NAVAL POSTGRADUATE SCHOOL
Monterey, California

THESIS

DESIGN OF AN ALGORITHM FOR MINIMIZING LORAN-C TIME DIFFERENCE ERROR

by

Frederick M. France, Jr.

September 1997

Thesis Advisor: Murali Tummala
Co-Advisor: Roberto Cristi

Approved for public release; Distribution is unlimited.

REPORT DOCUMENTATION PAGE

Form Approved OMB No. 0704-0188

1. AGENCY USE ONLY (*Leave blank*)	2. REPORT DATE	3. REPORT TYPE AND DATES COVERED
	September, 1997	Engineer's Thesis

4. TITLE AND SUBTITLE DESIGN OF AN ALGORITHM FOR MINIMIZING LORAN-C TIME DIFFERENCE ERROR

5. FUNDING NUMBERS

6. AUTHORS France, Frederick M., Jr.

7. PERFORMING ORGANIZATION NAME(S) AND ADDRESS(ES)
Naval Postgraduate School
Monterey CA 93943-5000

8. PERFORMING ORGANIZATION REPORT NUMBER

9. SPONSORING/MONITORING AGENCY NAME(S) AND ADDRESS(ES)
U.S. Coast Guard, Loran Support Unit
12001 Pacific Avenue, Wildwood, NJ 08260-3232

10. SPONSORING/MONITORING AGENCY REPORT NUMBER

11. SUPPLEMENTARY NOTES The views expressed in this thesis are those of the author and do not reflect the official policy or position of the Department of Defense or the U.S. Government.

12a. DISTRIBUTION/AVAILABILITY STATEMENT
Approved for public release; distribution is unlimited.

12b. DISTRIBUTION CODE

13. ABSTRACT(*maximum 200 words*)

The United States Coast Guard (USCG) is in the process of upgrading the hardware of the Loran-C Radionavigation System Control System. As part of this effort, the Computer-Assisted Loran-C Controller (CALOC), is also in need of improvement. CALOC performs four tasks: abnormality detection, time difference control, recordkeeping, and blink control. The work reported in this thesis focuses on time difference control. In many instances, CALOC does not accurately control the time difference error (TDE) within the established USCG control procedures. Two new algorithms are proposed here to control TDE more effectively: a proportional-integral-derivative (PID) controller and a Kalman filter. Actual TDE data recorded at three different master stations covering five Loran-C chains is used to evaluate the performance of the proposed controllers. The PID controller shows a substantial improvement in control compared to CALOC, and the Kalman filter exhibits even better performance, based on preliminary results. This improvement in control correlates directly with an increase in both predictable accuracy and repeatable accuracy.

14. SUBJECT TERMS
CALOC, Kalman filter, Loran-C, PID controller, Radionavigation

15. NUMBER OF PAGES 216

16. PRICE CODE

17. SECURITY CLASSIFICATION OF REPORT	18. SECURITY CLASSIFICATION OF THIS PAGE	19. SECURITY CLASSIFICATION OF ABSTRACT	20. LIMITATION OF ABSTRACT
Unclassified	Unclassified	Unclassified	UL

NSN 7540-01-280-5500

Standard Form 298 (Rev. 2-89)
Prescribed by ANSI Std. 239-18 298-102

DESIGN OF AN ALGORITHM FOR MINIMIZING LORAN-C TIME DIFFERENCE ERROR

Frederick M. France, Jr.
Lieutenant, United States Navy
B.S.E.E., University of Southern California, 1987

Submitted in partial fulfillment of the
requirements for the degrees of

MASTER OF SCIENCE IN ELECTRICAL ENGINEERING
and
ELECTRICAL ENGINEER

from the

NAVAL POSTGRADUATE SCHOOL
September 1997

ABSTRACT

The United States Coast Guard (USCG) is in the process of upgrading the hardware of the Loran-C Radionavigation System Control System. As part of this effort, the Computer-Assisted Loran-C Controller (CALOC), is also in need of improvement. CALOC performs four tasks: abnormality detection, time difference control, recordkeeping, and blink control. The work reported in this thesis focuses on time difference control. In many instances, CALOC does not accurately control the time difference error (TDE) within the established USCG control procedures. Two new algorithms are proposed here to control TDE more effectively: a proportional-integral-derivative (PID) controller and a Kalman filter. Actual TDE data recorded at three different master stations covering five Loran-C chains is used to evaluate the performance of the proposed controllers. The PID controller shows a substantial improvement in control compared to CALOC, and the Kalman filter exhibits even better performance, based on preliminary results. This improvement in control correlates directly with an increase in both predictable accuracy and repeatable accuracy.

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

ACKNOWLEDGMENTS

This thesis is dedicated to my father-in-law, Clarence A. Benson, Jr. (23 May 1936 - 30 May 1997), who is a significant reason for who I am today and continues to be a guiding force in my life.

I would like to thank my family, particularly my wife, Donna, and children, Annemarie, Laurel, Marguerite and Frederick, for their steadfast love and support. I would also like to thank my extended family in Monterey, especially the members of St. Mary's-by-the-Sea parish, whose prayers sustained me through this endeavor.

Thanks are also due to Professors Murali Tummala, Roberto Cristi, and John G. Ciezki at the Naval Postgraduate School for their assistance and guidance. Finally, LCDR Chuck Schue, USCG, and LT Steve Bartlett, USCG, for the great support I received from the United States Coast Guard's Electronics Engineering Center (EECEN), without which this research could not have been accomplished.

I. INTRODUCTION

As part of the 1996 Federal Radionavigation Plan, Loran-C is scheduled to be decommissioned on 31 December 2000, and the Global Positioning System (GPS) is to become the sole source of global radionavigation. However, there is still significant interest in the use of Loran-C for air, land and marine navigation, and additional concern about not having a back-up, or complementary, navigation system to GPS. Under the Coast Guard Authorization Act of 1996, the Department of Transportation is required to present a report on the impact of the Loran-C decommissioning. There are numerous factors that will influence the decommissioning decision, including international treaties and alliances, costs of maintaining more than one federally-funded radionavigation system, and domestic user requirements. Improving the accuracy of Loran-C and reducing the manpower required to operate the stations will lead to reduced operational costs, thus making Loran-C a feasible option to complement GPS.

Maintaining the technological edge of legacy systems has always been a challenge to design engineers. The United States Coast Guard's (USCG) efforts to improve the timing control of the Loran-C transmitted pulse is no different. In the late 1980s, the USCG embarked upon an aggressive multi-year project, titled "Electronics Equipment Replacement Project" (EERP), to upgrade the entire Loran-C system. As part of EERP, Coast Guard engineers at the Electronics Engineering Center (EECEN) have identified commercial-off-the-shelf (COTS) items that will replace outdated control subsystem hardware and consolidate personnel tasking.

To complement this hardware upgrade, the control algorithm needs to be improved. The existing system, the Calculator-Assisted Loran-C Controller (CALOC), does not maintain Loran-C within USCG control specifications. For example, in the recorded data provided by the USCG for use in this thesis, there are cumulative time difference errors (TDEs) in excess of 100 ns and instantaneous TDE exceeding the USCG established control limit of ± 50 ns. These large TDEs are directly related

to the inadequacy of the existing control algorithm. The Loran-C operators, knowing that CALOC is unreliable, do not allow it to automatically control the Loran-C chain. It is impossible to determine how the operators' actions influence the accuracy of CALOC.

Timing control of the Loran-C transmitted pulse is directly related to both the absolute accuracy and the repeatable accuracy of the Loran-C system. Both types of accuracy are important, but for different reasons. For example, the absolute accuracy is important to the mariner who may be entering an unfamiliar port for the first time while the repeatable accuracy is important to the coastal fisherman who lays lobster traps in the morning and wants to return to the same location in the evening. In general terms, the absolute accuracy of the Loran-C system includes both the random errors and the systematic errors while the repeatable accuracy includes only the random errors. The algorithm for control of the transmitted pulse timing is only capable of controlling the random errors. Hence, an improvement in control of the transmitted pulse timing leads directly to an improvement in the repeatable accuracy.

The goal of this work is to develop a new control algorithm that will maintain control of both the instantaneous and cumulative TDE within USCG specifications, and in turn improve operator confidence in the control system. The algorithm is developed using actual data received from the monitor stations and the output from the existing control algorithm. The data is first preprocessed to remove as much of CALOC's influence as possible to achieve uncorrected data. The preprocessed data retains the effects of atmospheric, terrain-related and man-made noise that cause random errors in the Loran-C signal. This simplifies the model required for implementing the algorithm, and it also enables a more accurate solution since the control algorithm is tested using field-recorded data.

In order to develop a new control algorithm, an understanding of Loran-C with an emphasis on CALOC is required. A brief overview of Loran-C is contained in

Chapter II, along with a description of the EERP as it applies to Control Equipment Consolidation. Chapter III contains pertinent information about CALOC, detailing its algorithm and the theory supporting it. In Chapter IV, a proportional-integral-derivative (PID) control algorithm is developed which leads to a data-dependent model discussed in Chapter V. Finally, conclusions and recommendations for further study are given in Chapter VI.

The appendices contain Matlab plots and the code used to generate them. The strip charts based on the recorded data from CALOC are contained in Appendix A. Appendix B contains the plots for the same data controlled by the PID controller, and Appendix C contains the plots for the Kalman filter. Finally, the Matlab code is included in Appendix D, beginning with a brief description of the code and definitions of functions and variables.

II. OVERVIEW OF LORAN-C

LORAN, an acronym for LOng RAnge Navigation, is a member of the family of radionavigation systems that provides geographic position-fixing data in the form of hyperbolic lines of position. It operates on the principle that radio waves travel at a constant velocity. The measurement of the difference in arrival time of radio pulses from two time-synchronized transmitting stations establishes a hyperbolic line of position. The intersection of two or more lines of position provides a fix, or location. [Ref. 1]

A. HISTORY

Loran-C can trace its roots back to a hyperbolic radionavigation system proposed by R. J. Dippy in 1937 and later implemented as the British *Gee* system in early 1942. During this period in the United States, the same principles of hyperbolic radionavigation were being researched by the MIT Radiation Laboratory. Spurred by the events of World War II, development of Loran proceeded rapidly in the United States with a chain of transmitters, later called *standard Loran* or *Loran-A*, that were operated by the USCG starting in January 1943. These initial stations, along with others operated by the Royal Navy and the Royal Canadian Navy, provided coverage over the North Atlantic convoy routes. Additional stations were constructed in the Aleutian Islands and central Pacific Ocean so that by the end of the war Allied military ships and aircraft enjoyed nighttime coverage of over 30% of the earth's surface. [Ref. 1, 5]

Following World War II, research sponsored by the Department of Defense (DoD) and other agencies explored a more accurate and longer range version of Loran. This research led to various improvements, including *low frequency (LF) Loran*, *Loran-B*, *CYCLAN*, and *CYTAC*. The Cycle Tactical (CYTAC) Bombing and Navigation System, consisting of a master and two slave transmitter stations, was the

most promising development to evolve. The CYTAC system allowed a single mobile receiver to determine its position by automatically measuring the difference in time of arrival of radio-frequency (RF) pulses from the three transmitters. [Ref. 5]

In 1957, the United States Navy identified a need for a long-range maritime radio navigation system for use with the ballistic missile submarines. However, the accuracy and range required by the Navy considerably exceeded the capabilities of the existing Loran-A equipment. Based on the results of the CYTAC testing, it was believed that the requirement for long-range navigation could be met by implementing the CYTAC concept. CYTAC was made operational in 1957 and placed under the control of the USCG, designating it Loran-C, in 1958. One of the significant improvements was using a lower frequency band of 90 to 110 kHz, as opposed to the 1.85 to 1.95 MHz of Loran-A, which provided a significant increase in range for Loran-C. Additional technical improvements also enabled more accurate geographic positioning. [Ref. 1, 5]

Today, Loran-C is a series of transmitter chains throughout the world, some operated by the USCG and others by host nations that provide highly accurate, 24-hour-a-day, all-weather radionavigation used for ocean and coastal navigation. In the United States it is the federally provided radionavigation system for the U.S. Coastal Confluence Zone (CCZ).[1] Loran-C is used primarily by mariners, both domestic and international, but is also widely used by civilian aircraft in the United States. [Ref. 1]

B. LORAN-C SYSTEM

The basic element of the Loran-C system is the loran chain. The chain consists of one master loran station and at least two secondary stations. Typical configurations for Loran-C chains are illustrated in Figure 1. The secondary stations are

[1]The CCZ is defined as the area seaward of a harbor entrance to 50 nautical miles offshore or the edge of the Continental Shelf (100 fathom curve), whichever is greater. [Ref. 1]

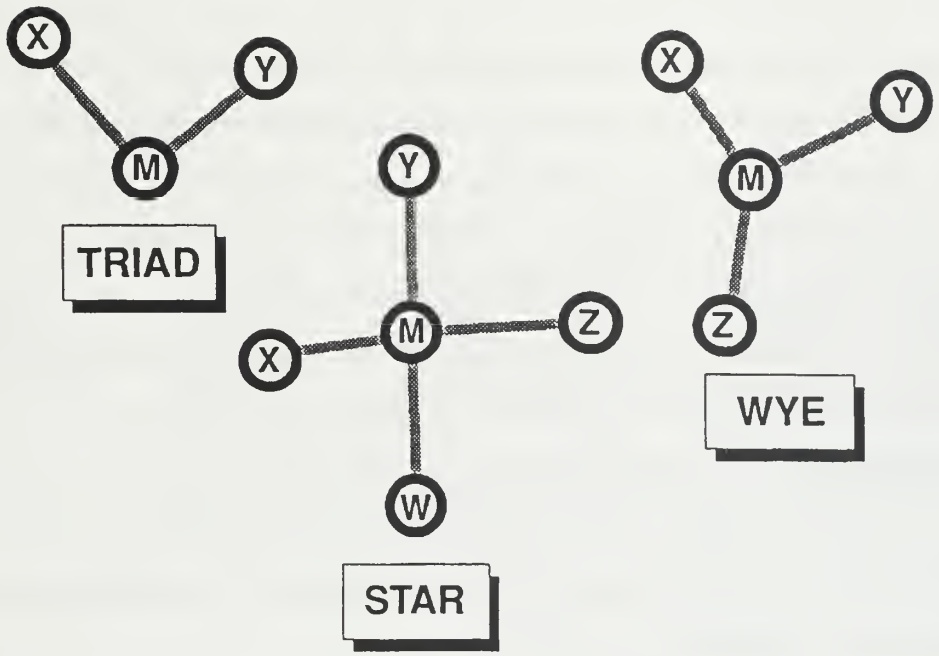


Figure 1. Typical Configurations for Loran-C Chains. [Ref. 1]

generally labelled starting with Zulu and working backwards through the alphabet. Each Loran-C chain provides signals suitable for accurate navigation over a coverage area. These coverage areas overlap in many parts of the United States and its coastal waters, where two or more chains may be received and used for navigation. [Ref. 1]

The concept of Loran-C is based on the time difference (TD) between the time of arrival (TOA) of the transmitted pulses of a master station and one of several secondary stations in a loran chain. Beginning with the master station, each loran station (LORSTA) within a chain transmits a series of phase-coded pulses. The timing of pulse transmission allows for each station's signal to propagate throughout the area of coverage before the next station transmits. Normally, the secondary stations transmit in alphabetical order (i.e., Victor, Whiskey, Xray, Yankee, and then Zulu). A receiver within the chain's area of coverage measures and displays the elapsed time between the signals from each station. The TD between the master and secondary

stations yields a hyperbolic line of position on which the receiver must lie. A fix[2] is determined by two or more intersecting hyperbolic lines of position. [Ref. 6]

The chain's master and secondary stations continue to sequentially transmit the phase-coded pulses with the master station transmitting a fixed time following the final secondary station in the chain. The length of time between each master station transmission is determined by the group repetition interval (GRI), which is assigned to each loran chain upon installation. Assigned GRIs range from 59,300 to 99,990 microseconds, which means that a chain's complete transmission cycle will repeat anywhere from 10 to 17 times each second. In order to ensure proper transmission timing, progressively longer emission delays[3] are assigned, so the transmission of the phase-coded pulses from each secondary station are allowed to propagate throughout the chain's area of coverage. [Ref. 6]

A typical transmission pulse pattern for a Loran-C chain, made up of a master station and three secondary stations, is shown in Figure 2. The emission delay of each secondary station, depicted as TDX, TDY and TDZ, is the length of time between the start of transmission for the master station and that of each secondary station. The GRI is illustrated as the time between the start of transmission for the master station and the next time it transmits. Also depicted in Figure 2 is an exploded view of the Loran-C pulse shape, consisting of sine waves enclosed within a teardrop shaped envelope, referred to as a t-squared pulse. [Ref. 1, 6]

C. THE ELECTRONIC EQUIPMENT REPLACEMENT PROJECT

In 1989, the United States Coast Guard (USCG) embarked on an effort to redesign portions of the Loran-C system aimed at taking it into the 21st century. The

---

[2]A fix is a known position determined the intersection of two or more lines of position (LOPs) determined from terrestrial, electronic and/or celestial data. [Ref. 1]

[3]Emission delay (ED) is the time difference, in μ seconds, between a master loran station transmission and a given secondary station transmission. [Ref. 1]

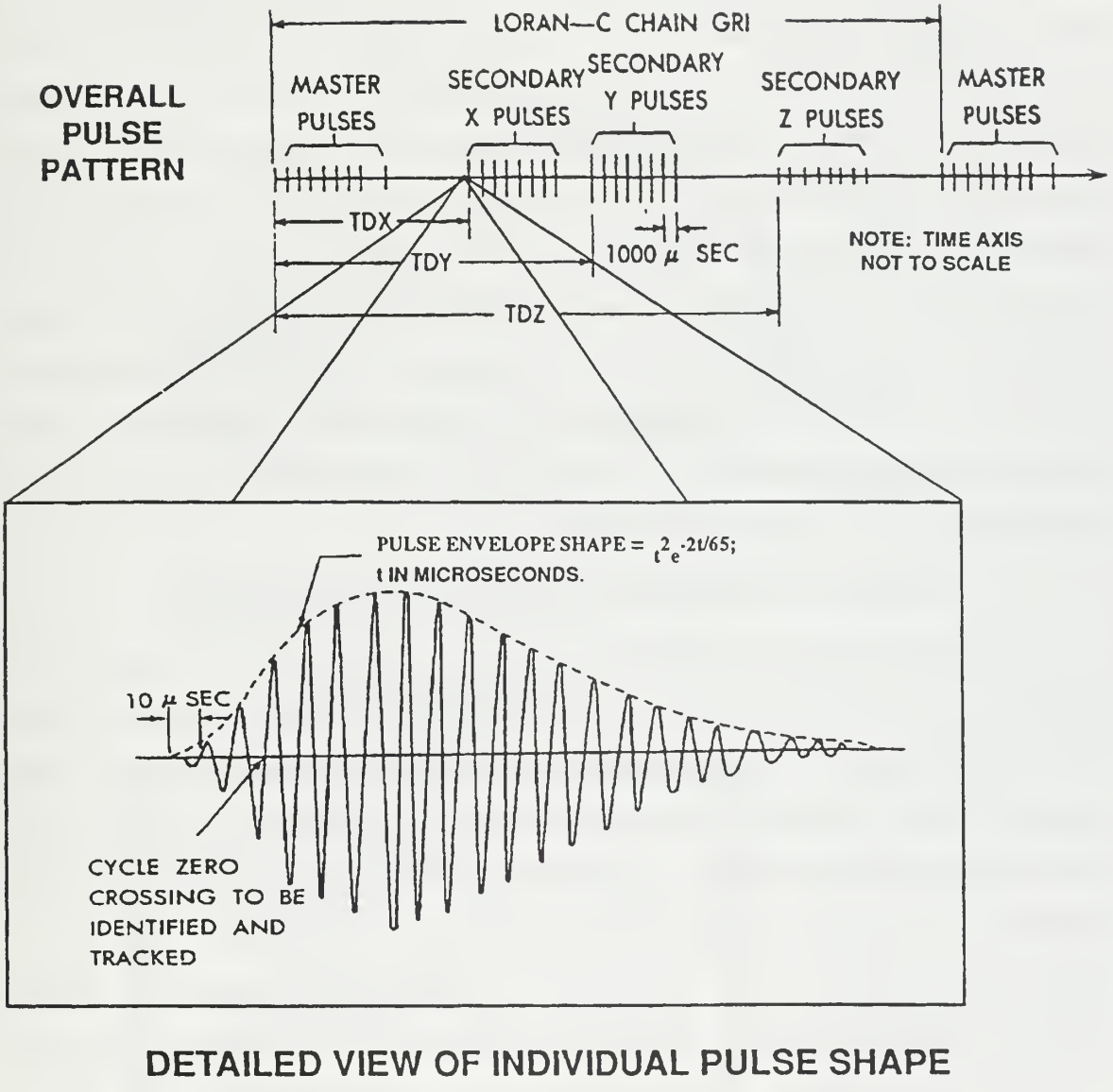


Figure 2. Pulse Pattern and Detailed Pulse Shape for Loran-C Transmission. [Ref. 1]

USCG's Electronics Engineering Center (EECEN) was tasked to perform a complete analysis of several subsystems under a multi-year project known as the Electronic Equipment Replacement Project (EERP) using the following criteria: the supportability of present day equipment; the support and maintenance of existing equipment projected ahead five years; the desire to enhance and expand automation; the need to respond to new system requirements; and the desire to remain in close step with technology. The primary purpose of EERP is to identify immediate support problems, develop near term solutions to these problems and chart the course for system improvements while addressing new requirements. Redesign of various portions of the Loran-C system is necessary to meet the demands of maintaining and operating the system into the next century. This thesis is concerned with analyzing the existing Loran-C Control Subsystem and developing an updated control algorithm to complement the hardware improvements being installed in support of the EERP's Loran Consolidated Control Station (LCCS). [Ref. 7]

1. Plan V - Control Equipment Consolidation

One of the goals of the EERP is to address a major flaw in the current control subsystem: excessive dependence on human watchstanders to evaluate incoming data from multiple sources. Currently, watchstanders constantly receive and must then understand both visual and audio outputs from four distinct pieces of equipment. Watchstanders have no method to interpret the received information unless they can correlate the data with information from another piece of equipment. Figure 3 illustrates the numerous inputs required for the watchstanders to interpret. While reducing the need to manually correlate information is not an adequate solution, all the information should be consolidated into one piece of equipment. This consolidation would significantly simplify the watchstander's job. Figure 4 outlines the various modules of the control subsystem. As part of the redesign, the input/output relationships shown in the figure would remain unchanged. [Ref. 7]

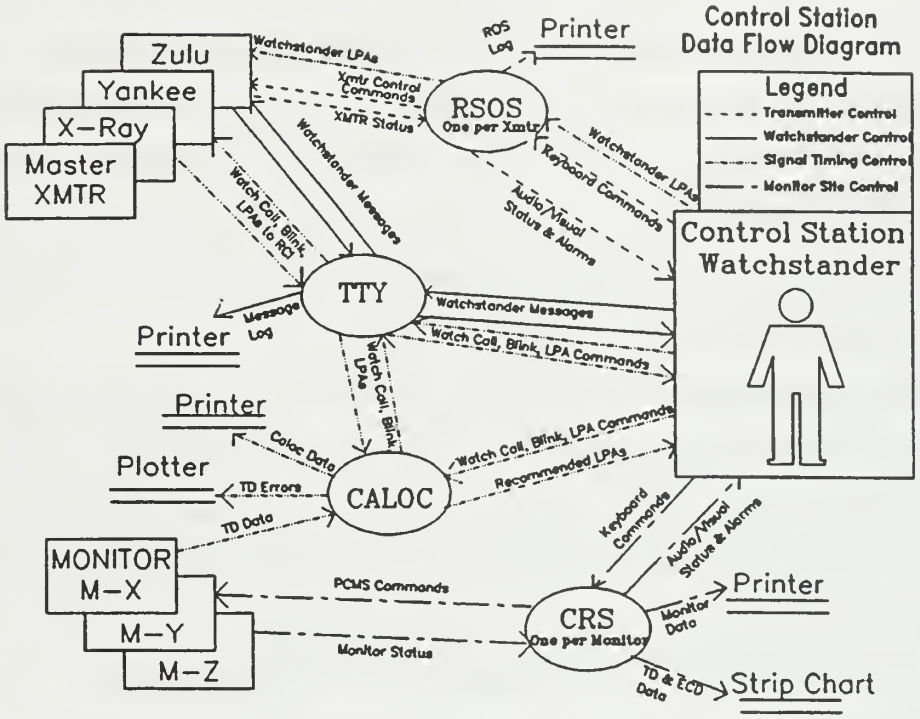


Figure 3. Data Flow Diagram Control Subsystem. [Ref. 2]

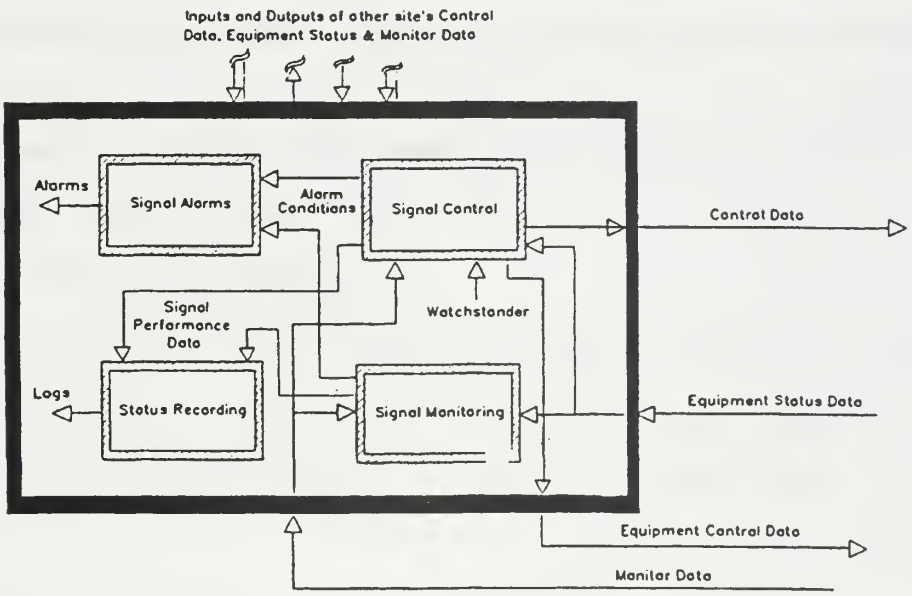


Figure 4. Control Subsystem Input/Output Relationship. [Ref. 3]

2. LORAN-C Consolidated Control Station (LCCS)

The requirements analysis associated with the Control Equipment Consolidation concluded that the watchstander is presented with too many sources of Loran chain status information and that this information must be consolidated into a single view of the Loran chain. The purpose is to generate this single view of the Loran environment and display it for the watchstander. The outgrowth of this project is the Loran Consolidated Control Station (LCCS).

The primary mission of the LCCS is to monitor and control LORSTAs and Primary Chain Monitor Sets (PCMS) currently functioning as part of the Loran-C Radionavigation System. The secondary mission is to replace the current suite of chain control equipment, which includes the Remote Site Operating Set (RSOS), Calculator-Assisted Loran-C Controller (CALOC), Chain Recorder Set (CRS) and Teletype (TTY). The chain control equipment at the three Continental United States (CONUS) Loran Control Stations in Malone, Florida, Seneca, New York, and Middletown, California, will be replaced. The LCCS will reduce and automate the data collection and decision-making functions at these stations, so they may be remotely controlled from central control sites in Alexandria, Virginia, and Petaluma, California. [Ref. 2]

The hardware and software components of the LCCS are divided into several Hardware Configuration Items (HWCI) and Computer Software Configuration Items (CSCI). Figure 5 depicts a top-level LCCS block diagram illustrating these configuration items. The LCCS Tactical Advanced Computer (TAC-4) is an HP 9000 workstation with an HP-UX (Unix-based) operating system. Two CSCIs will be used: one is the executable software to run LCCS, and the other is the network management software. Other HWCIs are the Data Retrieval/Monitoring System and the Communications Interface. [Ref. 2]

In concert with the implementation of Control Equipment Consolidation and installation of the LCCS, the LORSTAs will make a significant change in the method

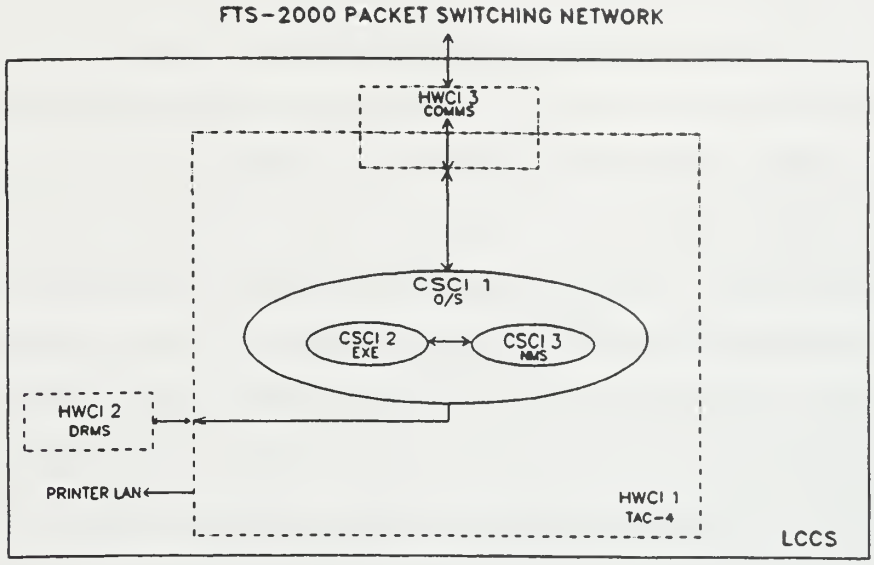


Figure 5. LCCS Top-Level Block Diagram. [Ref. 2]

of communicating between stations. The current process involves using either dedicated or leased telephone lines. With the upgrade, communications will employ X.25 data lines and packet assembling/disassembling equipment at all CONUS LORSTAs and PCMS sites. In addition to providing potentially significant cost savings, the packet switching system is able to be configured from any location. This will allow for monitoring and troubleshooting of the Loran-C communications network from a single location, such as EECEN. [Ref. 2]

3. Proposed Effort

One of the areas of the Control Subsystem that is not currently scheduled to be upgraded is the control algorithm resident in CALOC. The Coast Guard intends to port the current CALOC algorithm to the new Unix environment of LCCS with minimal functional change. The primary goal of this thesis is to investigate alternate solutions that will replace the existing algorithm. The new control algorithm must keep both the instantaneous time difference error (TDE) within ± 50 ns and the cumulative TDE close to the controlling standard time difference (CSTD) while

providing the minimum number of local phase adjustments (LPA), which are transmitter timing adjustments issued in multiples of 20 ns. The basic control law that will be followed is to minimize the TDE by maintaining the time difference as close as possible to its control number while minimizing the magnitude and number of timing adjustments applied. [Ref. 8]

In this chapter, a brief history and system description of Loran-C was presented. It also outlined the EERP, focusing primarily on Control Equipment Consolidation, and in particular the LCCS. As part of EERP, a new control algorithm needs to be developed to replace CALOC. In order to develop a suitable algorithm, an understanding of the existing system is required. The next chapter discusses the functionality of CALOC, including the four general tasks it performs. It also describes the control policies and the role of monitor stations in the performance of Loran-C.

III. CALCULATOR ASSISTED LORAN-C CONTROLLER

The Calculator-Assisted Loran-C Controller (CALOC) consists of a control algorithm implemented as a desktop calculator and various hardware peripherals, which may be installed at a Loran-C time difference control station. Its function is to relieve the operator of many tedious and routine jobs and perform several computations. CALOC monitors and controls the phase timing of the Loran-C signal on up to four baselines while the envelope timing is presumed to be monitored and maintained at each individual transmitting station via the installed local signal envelope-to-cycle difference (ECD) measurement, limit check, and alarm circuits. [Ref. 8]

A. TASKS

In order to monitor and control the Loran-C system, CALOC has been designed to perform four general tasks: abnormality detection; time difference (TD) control; recordkeeping; and blink control. In performing these tasks, CALOC acts mainly to assist, not replace, the human operator, especially in the area of subjective judgment which must be applied during many system casualty conditions. [Ref. 8]

1. Abnormality Detection

An equipment casualty at either the master or the secondary transmitting station can cause a sudden excursion, or abnormal behavior, in the received phase timer difference (TD). Since CALOC cannot possibly anticipate the symptoms of, and corrections for, all possible system casualties, it simply inhibits control task operation on a baseline when it finds that the baseline is abnormal. At the same time, CALOC notifies the human operator of the abnormal condition and then relies upon the operator to correct it. When the operator is satisfied that the baseline is again normal, he returns control to CALOC. [Ref. 5, 8]

To determine if a baseline is abnormal, CALOC uses a statistical algorithm. The CALOC system, in the face of continuously changing received noise power, continually evaluates the two hypotheses — baseline normal or baseline abnormal — while attempting to satisfy the conflicting goals of minimizing the time to detect and the probability of making an error. In this manner, CALOC relieves the human operator of the need for constant attention to his receiver displays and strip charts. [Ref. 5, 8]

2. Time Difference (TD) Control

Under steady-state or normal conditions, the time difference will fluctuate due to the transmitter oscillator activity and other transmitter station variations, and propagation variations. Because the received signal is contaminated by noise, the optimum number and size of LPAs necessary to keep the time difference at the established control number is not obvious, and it is difficult for a human operator to properly weigh all of the many factors in the decision to insert an LPA. [Ref. 5, 8]

The most important function of CALOC is to quantify the process of observing the present and past time difference behavior, weigh the competing factors involved in the insertion of an LPA, and choose the optimum correction to insert. This system transmits the optimum LPA value to the appropriate Remote Control Interface (RCI) for insertion at the remote station when operating in the automatic control mode. In the semi-automatic control mode, the system recommends that the operator make the necessary timing correction. The operator is allowed the option of inserting the LPA manually, allowing CALOC to insert the LPA, or aborting the recommendation altogether. In this way, the operator has the final decision on the LPA when CALOC is placed in the semi-automatic control mode. If the CALOC system is initialized for a default manual mode, recommendations will be made, but CALOC will not transmit the RCI control messages. The manual mode is provided for those control stations that have unreliable teletype communications. [Ref. 5, 8]

3. Recordkeeping

In the execution of the above tasks, CALOC generates many numerical values which are needed by the Coordinator of Chain Operations (COCO) for the proper management of the Loran-C system. Since these values can assist the TD control station personnel in the preparation of their operational messages, CALOC writes them onto a printer tape and drives a plotter to show exactly how the baselines under its control performed. By using the printer tape and plotter output, which CALOC annotates with the day's major events, the operator is relieved of much of the recordkeeping burden. [Ref. 5, 8]

4. Blink Control

Blink codes are used to transmit the status of Loran-C system accuracy and the reliability of the stations within the chain. If one of the stations is out of tolerance or unusable for some reason, the affected station will blink. Blinking is a repetitive on-off pattern of the first two pulses of the secondary signal, which indicates that the baseline is unusable for one of the following reasons:

- Time Difference out of tolerance,
- Envelope-to-Cycle Difference out of tolerance,
- Improper phase code or Group Repetition Interval, or
- Master or secondary station operating at less than one-half of specified output power, or master station is off-air.

When any of the above conditions occur, the operator will initiate Blink via CALOC. [Ref. 5]

B. CONTROL POLICIES

The Coast Guard's stated objectives in controlling the TD of a Loran-C baseline at the primary control monitor are to use a limited number of LPAs while keeping the TD close to its assigned number as well as keeping the cumulative TDE close to

zero. Every 15 minutes, CALOC determines whether an LPA is required to correct the time difference. To convert the evaluation or determination of whether or not to insert an LPA into a numerical procedure suitable for CALOC, a control cost is defined. This cost, J, is a weighted sum of the deviation of the time difference from the assigned number, the deviation of the cumulative TD error from zero, and the size of the LPA being considered:

$$J(u) = \frac{(Z_N - N_C + u)^2}{K_s^2} + \frac{\epsilon^2(u)}{K_i^2} + \frac{u^2}{(20ns)^2} \tag{3.1}$$

where Z_N is the most recent TD estimate, u is the value of the individual LPA being evaluated, $\epsilon(u)$ is the total predicted cumulative error, $(Z_N - N_C)$ is our predicted TDE, and K_i and K_s are parameters used to alter the controller function. The cost is computed using a least squares method for each possible LPA. The LPA which results in the least cost is then selected. [Ref. 5, 8]

In order to assist in minimizing the number of LPAs during times of high atmospheric noise, Loran uses two different control policies (CONPOL). Two sets of weighting factors, K_i and K_s, associated with CONPOL 1 and CONPOL 2, respectively, are stored in CALOC and used to provide adaptive properties to the algorithm. The time of their application is controlled by the COCO. One use of the control policies is to provide close control during the day and loose control at night when there is, in general, more noise in the atmosphere. [Ref. 5, 8]

C. MONITORING STATIONS

The purpose of an unmanned remote Loran-C monitor site is to receive signals from Loran-C transmitting stations and provide regular reports on cycle time difference, absolute envelop-to-cycle difference (ECD), signal-to-interference ratio, and signal strength to the respective chain control sites. The chain control sites use the monitor information to hold Loran-C transmitting station signals to within assigned tolerances. Each Loran-C monitor site consists of a receiving whip antenna with a

wire ground plane and a small, all-weather shelter, which houses the Loran-C monitor equipment and attendant prime power and communication service tie points. There are two primary pieces of equipment that the monitoring station uses to process the received Loran-C signals: the Austron 5000 and the PDP-8/E computer.

1. Austron 5000

The Austron 5000 is a high accuracy receiver used as a remote sensor for control of the Loran-C chain. The receiver may be used on a time-shared basis to monitor and track up to four chains consisting of eight stations. The system consists of a receiver section which filters and amplifies the Loran-C signal and then digitizes samples upon command from the computer section. The receiver section is initialized and controlled by a PDP-8/E general purpose 12-bit minicomputer. All processing is done in digital servo loops and filters written in the monitor program. [Ref. 5]

The receiver uses direct RF sampling. During track mode, three samples are taken on each pulse: one zero crossing sample at 30 μs called the phase strobe; one at +2.5 μs from the phase strobe called the amplitude strobe; and one at -7.5 μs from the phase strobe called the cycle strobe. Additionally, a guard sample is taken from the pulse on a time-shared basis to check for early energy. These three sample and hold circuits are followed by a 12-bit analog-to-digital (A/D) converter which sequentially digitizes the three samples and passes them to the PDP-8/E computer for processing in software phase-locked loops and filters. [Ref. 5]

2. PDP-8/E Computer

Each monitor station has two PDP-8/E computers: one is used to control the Austron 5000, and the other is maintained in a standby condition. The in-service computer treats the receiver section, including the Austron 5000, as a peripheral device and collects TD data, envelope data, receiver gain and noise data from the Austron 5000. This information is formatted by the computer into the data messages,

coded strip chart update messages, and error messages. These messages are then sent to the control monitor station via a patch panel and modem. [Ref. 5]

D. TIME DIFFERENCE ERROR (TDE)

The time difference error (TDE) is the principal parameter used to control the timing of the transmitted Loran-C pulse. There are two components that make up the TDE: the controlling standard time difference (CSTD) and the time difference (TD). The CSTD is established through a chain calibration process, of which the primary purpose is to ensure that the emission delay (ED) of each secondary station is set to the value published by the USCG. The time difference is a measurement taken at the monitor station and is used to determine the time difference error (TDE). [Ref. 9]

1. Controlling Standard Time Difference (CSTD)

The CSTD for each master-secondary pair is established by the Coast Guard after a secondary station has been installed and the chain is ready to become operational. The process to obtain the CSTD begins when the Coast Guard calibrates a cesium beam frequency standard, with an optional time standard, at the United States Naval Observatory (USNO) or the National Institute of Standards and Technology (NIST). The cesium time standard is operated for about a month, tuning it so that it will match the master clock, or universal time constant (UTC), at USNO. The Coast Guard also tries to minimize the drift of the cesium time standard. [Ref. 9]

Once the cesium time standard is calibrated, the Coast Guard installs it in a master station to use as a time standard. The cesium time standard is compared to the time of transmission (TOT) of the master station. The TOT is adjusted to a pre-calculated value that is based on the group repetition interval (GRI) and the baseline length. Once the TOT is determined, the cesium time standard is brought to each of the secondary transmitter sites and connected similarly to that at the master station. The secondary TOT is then adjusted to its pre-calculated TOT value. [Ref. 9]

During chain calibration, the emission delay (ED) is calculated from the difference between the master station TOT and the secondary station TOT:

$$ED_S = TOT_S - TOT_M \tag{3.2}$$

where the subscripts S and M denote secondary and master, respectively. Once the emission delay is within specification, a correction to the CSTD is calculated:

$$COR = ED_P - ED_S \tag{3.3}$$

where ED_P is the published ED for the baseline. The correction is then added to the presently assigned CSTD to calculate the value for the new CSTD. This value then becomes the CSTD for a particular master-secondary baseline. CSTDs only exist for A1 monitor sites. The A2 monitor sites do not have a CSTD. [Ref. 9]

2. Phase Time Difference

The phase time difference (TD) is the difference between the TOA of a master station's signal and the TOA of a secondary station's signal from the same group repitition interval (GRI). Continuous measurement of the TD of each respective master-secondary pair is made at a monitor location, or locations, within the defined service area. This TD is maintained at the CSTD by the insertion of LPAs. In general, the hourly average of TD is held to within ± 50 ns of the CSTD.

E. CALOC PROJECT

1. Background

The Loran-C control problem is one of controlling a TD, specifically, the difference between the times of arrival (TOA) of a master station and a secondary station signal. In a study conducted in the mid-1970s, data was collected at the USCG Electronics Engineering Center (EECEN) over three unique propogation paths. The path from EECEN to the M station was nearly all overland, coastal direct path;

Y station was entirely over water; and Z station contained mid-western plain and the Appalachian mountains through Pennsylvania. In order to study the nature of the short term time difference (TD) fluctuations, as well as model them, TOAs were measured for M-Y and M-Z. This process enabled the TD to be measured more accurately since the TOA for each station was compared to its local cesium oscillator. [Ref. 4]

The TOA records used to develop the model that became the existing controller were generated from late October to December 1974. Correlation analysis performed on the data sets determined that a slowly-varying, long-term component was present in the data. After determining that the source of the long-term correlation was due to a long-term TOA fluctuation, it was approximated as a linear trend and removed using linear regression. The resulting autocorrelation with the linear trend removed exhibited a more expected behavior. The significant result of the TOA correlation analysis was that the TOA fluctuation is composed of two components: a short-term component with correlation of less than two hours and an rms value on the order of 20-30 ns, and a long-term component that appears as a linear trend over periods of 12-24 hours with a slope on the order of 10-20 ns/hour. [Ref. 4]

Two experiments were conducted to attempt to isolate the possible sources of the short-term TOA fluctuations. Due to the propagation effects being distributed along lengthy baselines, it was expected that the fluctuations would be due to transmitted signal synchronization. One source of fluctuation was the local cesium oscillator, which maintains real-time synchronization in each LORSTA. The second major source of fluctuation was the cycle-compensation servo loop, which measures the pulse crossover timing against a reference derived from the oscillator. The source of the fluctuation was the 12-second servo loop time constant, which provided the fastest transient recovery and also introduced the greatest amount of noise due to the random-phase-averaging measurement technique. [Ref. 4]

An attempt was made to quantify the fluctuations. It was determined that the primary source of TOA fluctuations were the random phase noise of the local cesium oscillators with a magnitude on the order of 7-14 ns. Additionally, the cycle-compensation servo loop did not contribute a statistically significant amount to TOA fluctuation. However, the servo loop and other unknown effects did make up some portion of the observed fluctuations. [Ref. 4]

2. Models

From the analysis conducted in the EECEN study as outlined above, it was determined that the Loran-C signal TOAs and, correspondingly, the TDs displayed fluctuations having two distinct characteristics: a short-term fluctuation with minimal correlation beyond the five minute sampling period used during the study; and a long-term fluctuation which appears as a linear trend over a 12-24 hour period. The data was insufficient to determine the source of the long-term fluctuation. This was deemed unimportant since control of the TDs would be over a much shorter period. Two models were explored to provide TD control: a minimum linear model and a more complex curve-fit and smooth approach. [Ref. 4]

a. Linear Model

The minimum linear model, which accounted for only the short term TOA fluctuation, did not make use of any smoothing or averaging techniques. It simply used the current sample value as the present best estimate for use in determining the correct control signal. The model was inadequate for several reasons: it did not recognize the contribution of the long-term TOA fluctuation nor the rapid changes in received signal-to-noise ratio (SNR) that may occur due to heavy weather activity; it was not desired to generate a control signal in quanta of 20 ns every sample period; and it would not adequately control the TD for received SNR conditions significantly worse than the observations recorded for this study. [Ref. 4]

The limitations of a minimum linear model meant that the control algorithm would not make a major reduction of the short-term TOA fluctuation since

that would lead to an undesireable return to fully synchronized operation. Thus, a short-term TD fluctuation of 20-30 ns was the best that could be attained with the existing equipment and a new algorithm. In order to include the long-term fluctuations, a model consisting of an integrator driven by white noise with a pole on the unit circle at z=1 was used. [Ref. 4]

This estimator of TOA fluctuation was a simple form of Kalman filter, as depicted in Figure 6. The value for K was determined empirically from the recorded TOA values. However, the value of K could have been determined for either a transient or a steady-state condition given the variance of the process driving the integrator and the variance of the received atmospheric noise. These parameters represent the time-varying nature of the TD control problem, and therefore a unique value could not be determined. [Ref. 4]

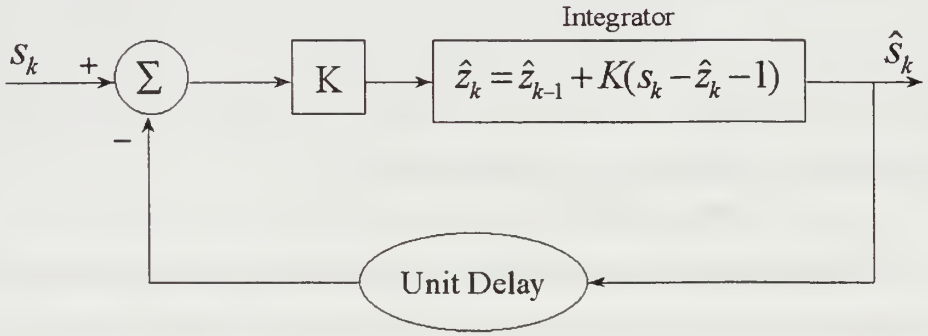


Figure 6. Estimator for Single Integrator Source Model. [Ref. 4]

b. Curve-fit Model

An alternative approach to the TD control problem explored a form of curve-fitting the previous TOA samples with an n^{th} order polynomial. There were several parameters considered in selecting the proper curve-fit model: the fit interval, the type and degree of curve to fit, and the fitting method. The biggest advantage of curve fitting was that it required no *a priori* knowledge of the TOA fluctuation process or the additive received noise, the parameters that were highly variable with time and location. [Ref. 4]

The choice of a 3rd order polynomial fit over several hours was purely subjective, based on qualitative observation and the observer's judgment and experience. The quadratic curve-fit of the TOA data provided one additional degree of freedom over its generally linear representation. The selected length of time over which to smooth the TOA data was $1\frac{1}{2}$ hours, or 18 data points. The 3rd order polynomial fit provided an offset, linear trend, and a quadratic curve to smooth the TD data. The least squares approach taken was expressed in matrix form as:

$$\mathbf{S} = \left[\begin{array}{cccc} S_1, & S_2, & \cdots, & S_N \end{array} \right]^T \tag{3.4}$$

$$\mathbf{Z} = \left[\begin{array}{cccc} Z_1, & Z_2, & \cdots, & Z_N \end{array} \right]^T \tag{3.5}$$

$$\mathbf{\Pi} = \left[\begin{array}{ccc} 1 & 1 & 1 \\ 1 & 2 & 4 \\ \vdots & \vdots & \vdots \\ 1 & N & N^2 \end{array} \right] \tag{3.6}$$

$$\mathbf{Z} = \mathbf{\Pi} \left[\begin{array}{cc} \mathbf{\Pi} & \mathbf{\Pi}^T \end{array} \right]^{-1} \mathbf{\Pi}^T \mathbf{S} \tag{3.7}$$

where $\mathbf{S}$ is an array of received TD samples with S_N the most recent sample; $\mathbf{Z}$ is an array of smoothed TD estimates with Z_N the most recent estimate; and $\mathbf{\Pi}$ is an $N \times 3$ third order polynomial matrix. [Ref. 4]

F. CALOC ALGORITHM

The comparison of the linear model and the curve-fit model failed to definitively prove that one was superior to the other. The curve-fit model was selected based on the improvement in using smoothed data versus real-time estimates and its superior adaptability to changing TD fluctuation and atmospheric noise conditions. The superior adaptability is primarily a consequence of the model being non-recursive: any transient disturbances beyond the smoothing interval have no effect while a recursive filter, like the linear model, has an infinite memory to these transients. [Ref. 4]

1. Performance Metric

The USCG has established control procedures for how the Loran-C system will be operated. These control procedures may be summarized as:

1. Insert LPAs to maintain the current time difference error (TDE) at the control station to $\pm\frac{1}{2}$ of the tolerance bound, which is nominally $\pm$ 100 ns;
2. Minimize the number of LPAs; and
3. Maintain the cumulative TD close to the CSTD.[Ref. 4]

The rationale behind these procedures is related to the accuracy and the stability of the Loran-C system. The predictable accuracy[1] is enhanced by maintaining the current TDE near zero, and the repeatable accuracy[2] is enhanced by maintaining the cumulative TD average at the CSTD. [Ref. 1] Minimizing the number of LPAs will provide a more stable system by not responding immediately to instantaneous fluctuations in noise level due to rapidly changing atmospheric conditions or other interference.[Ref. 4]

2. Cost Function

In order to be able to quantify the accuracy of any algorithm, a control law must be defined. The Loran-C control procedures were translated into a cost function:

$$J(u) = \frac{(Z_N - N_C + u)^2}{K_s^2} + \frac{\epsilon^2(u)}{K_i^2} + \frac{u^2}{(20ns)^2} \tag{3.8}$$

where u is the value of the individual LPA being evaluated, N_C is the control number, and $\epsilon(u)$ is the total predicted cumulative error. Qualitatively there is a cost associated with allowing the TD to deviate from the control number, N_C, and there is also a cost to correct the error with an LPA. Similarly, there is a cost associated

[1]Predictable accuracy is the accuracy of a position with respect to the geographic or geodetic coordinates of the earth. [Ref. 1]

[2]Repeatable accuracy is the accuracy with which a user is able to return to a known position whose coordinates have been measured at a previous time with the same navigational system. [Ref. 1]

with the cumulative TDE. The third term in the cost function is a penalty for using large LPA corrections. [Ref. 4]

The terms K_s and K_i are parameters used to tune the control functionality. Reducing the value of K_s causes any TD deviation from N_C to be costly while reducing K_i places the control emphasis on maintaining the cumulative TDE near zero. Increasing both K_s and K_i will cause the cost of an LPA to be increased such that few LPAs will be recommended until either the current TDE, $(Z_N - N_C)$, or cumulative TDE becomes excessively large. [Ref. 4]

In this chapter, a brief description of CALOC was presented with a focus on the task of time difference control. The significant portions of the original CALOC study were presented, including the existing algorithm and associated cost function. The next chapter will explore the proportional-integral-derivative (PID) controller. The development process of the PID controller includes preprocessing the recorded data and testing the algorithm. This, along with the results of testing, will be presented in the next chapter.

IV. PROPORTIONAL-INTEGRAL-DERIVATIVE CONTROL ALGORITHM

In this chapter, an algorithm based on the proportional-integral-derivative (PID) controller is developed. The input is actual data from operational Loran-C chains and the output values are local phase adjustments (LPAs). These LPAs are used to help maintain the station pulse transmission timing in line with the USCG established control procedures. The data provided is preprocessed in order to remove the influence of the current controller, CALOC, to obtain uncontrolled data. The data is filtered to remove the high frequency noise components and then passed through the PID controller. The output of the controller is quantized to produce LPAs in quanta of 20 ns for pulse timing correction. The results of this controller show improved control of both the current time difference error (TDE) and the cumulative TDE over the existing control algorithm.

A. TIME DIFFERENCE ERROR (TDE) DATA SETS

The data sets used in this research were collected during three distinct times of the year. The data from Malone, Florida, was collected during the winter; data from Middletown, California, was collected during the late spring; and the data from Kodiak, Alaska, was collected during the fall. This gave some indication of seasonal weather effects, however, no conclusions could be drawn from the limited data sets and the fact that each chain's data set included at most two days' worth of data.

1. Data Set Locations

There are eight Loran-C chains in North America that are operated by the USCG: North Pacific (NORPAC); Gulf of Alaska (GOA); U.S. West Coast (USWC); North Central U.S. (NOCUS); South Central U.S. (SOCUS); Great Lakes (GLKS); South-East U.S. (SEUS); and North-East U.S. (NEUS). The data used in this research encompasses five of the eight chains, though the data for NORPAC was recorded on

one day for only 15 hours. In the following discussion, the chain locations will be characterized, and the required preprocessing will be described.

a. Malone, Florida

Data collected from Malone, Florida was recorded from two chains: South-East United States (SEUS) and South-Central United States (SOCUS). It was recorded over a three-day period during 18-20 January 1997, though only the data from 18 and 20 January were used due to corruption of the 19 January data. The SEUS chain has a master station and four secondary stations with locations listed in Table I. The M-X baseline is primarily over water across the Gulf of Mexico while the other three baselines (M-W, M-Y and M-Z) are over coastal land masses. The estimated groundwave coverage area is depicted as the dashed line in Figure 7.

Station	Location
M	Malone, Florida
W	Grangeville, Louisiana
X	Raymondville, Texas
Y	Jupiter, Florida
Z	Carolina Beach, North Carolina

Table I. South-East United States Loran-C Chain Transmitter Locations. [Ref. 1]

The SOCUS chain has a master station and five secondary stations with locations listed in Table II. All five master-secondary baselines are over land, generally in the central plains area of the United States. The estimated groundwave coverage area is depicted as the dashed line in Figure 8.

b. Middletown, California

The data collected from Middletown, California was recorded on 4-5 May 1997 from two chains: United States West Coast (USWC) and North-Central United States (NOCUS). The USWC chain has a master station and three secondary stations with locations listed in Table III. The master-secondary baselines are all over land in the Rocky Mountain range of the western United States. The estimated groundwave coverage area is depicted as the dashed line in Figure 9.

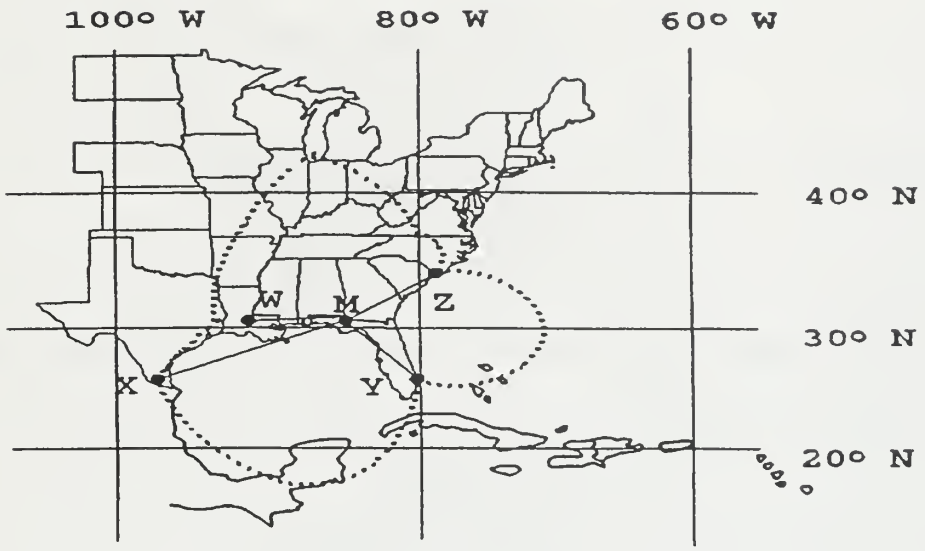


Figure 7. South-East United States Loran-C Chain. Transmitter Stations: M - Malone, FL; W - Grangeville, LA; X - Raymondville, TX; Y - Jupiter, FL; Z - Carolina Beach, NC. [Ref. 1]

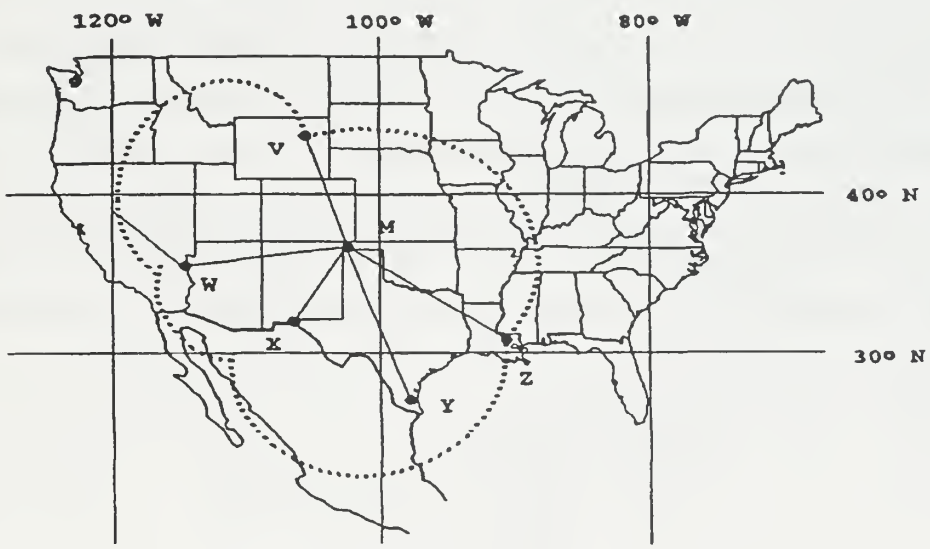


Figure 8. South Central United States Loran-C Chain. Transmitter Stations: M - Boise City, OK; V - Gillette, WY; W - Searchlight, NV; X - Las Cruces, NM; Y - Raymondville, TX; Z - Grangeville, LA. [Ref. 1]

Station	Location
M	Boise City, Oklahoma
V	Gillette, Wyoming
W	Searchlight, Nevada
X	Las Cruces, New Mexico
Y	Raymondville, Texas
Z	Grangeville, Louisiana

Table II. South Central United States Loran-C Chain Transmitter Locations. [Ref. 1]

Station	Location
M	Fallon, Nevada
W	George, Washington
X	Middletown, California
Y	Searchlight, Nevada

Table III. United States West Coast Loran-C Chain Transmitter Locations. [Ref. 1]

The NOCUS chain also has a master station and three secondary stations with locations listed in Table IV. The master-secondary baselines in this chain are located in the northern plains region of the United States and southern Canada. The estimated groundwave coverage area is depicted as the dashed line in Figure 10.

c. Kodiak, Alaska

The data collected from Kodiak, Alaska was recorded on 11 September 1996 for a period of 15 hours from Loran-C Station Zulu of the North Pacific

Station	Location
M	Havre, Montana
W	Baudette, Minnesota
X	Gillette, Wyoming
Y	Williams Lake, Canada

Table IV. North Central United States Loran-C Chain Transmitter Locations. [Ref. 1]

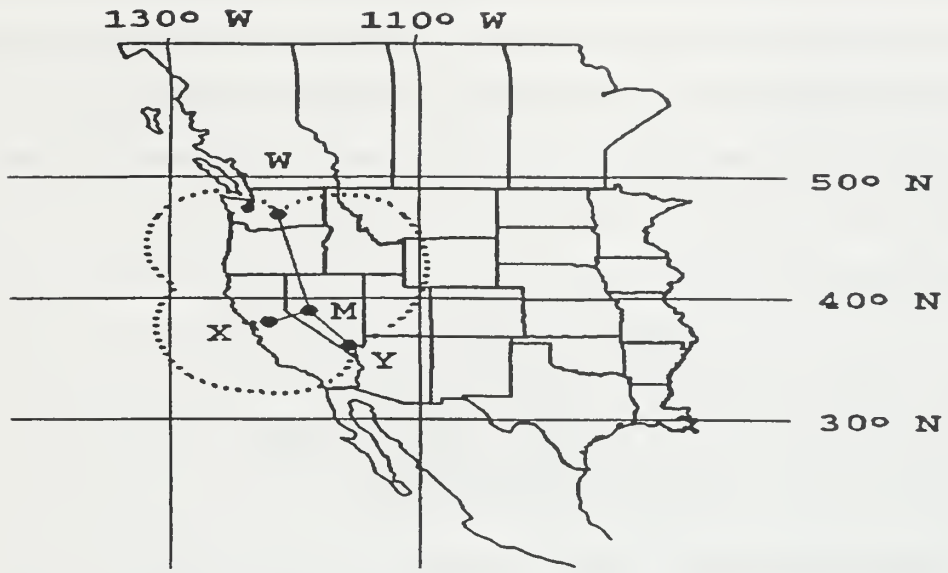


Figure 9. United States West Coast Loran-C Chain. Transmitter Stations: M - Fallon, NV; W - George, WA; X - Middletown, CA; Y - Searchlight, NV. [Ref. 1]

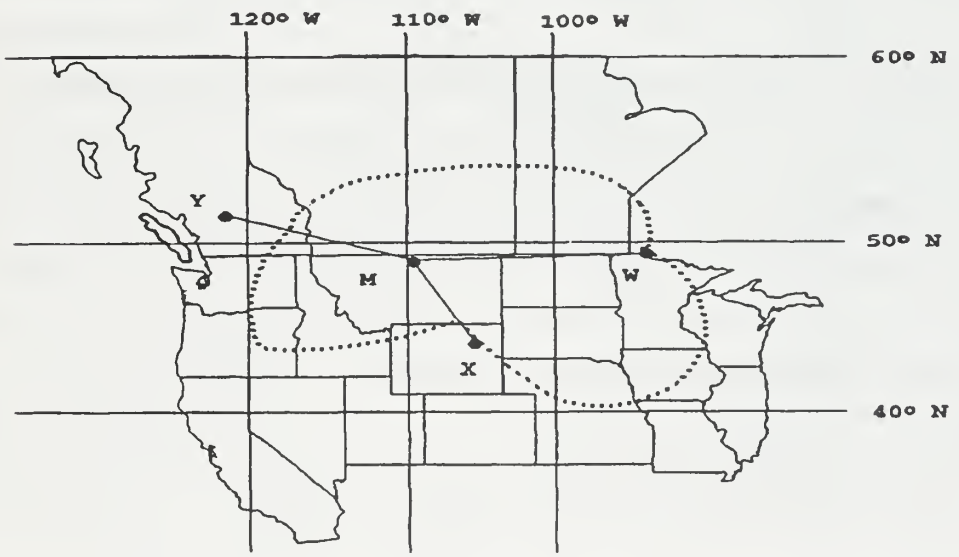


Figure 10. North Central United States Loran-C Chain. Transmitter Stations: M - Havre, MT; W - Baudette, MN; X - Gillette, WY; Y - Williams Lake, Canada. [Ref. 1]

(NORPAC) Chain. The NORPAC chain has a master station and three secondary stations with locations listed in Table V. The master-secondary baselines are primarily over water in the Bering Straits, though the M-Z baseline passes over the Alaskan peninsula. The estimated groundwave coverage area is depicted as the dashed line in Figure 11.

Station	Location
M	St. Paul, Alaska
X	Attu, Alaska
Y	Port Clarence, Alaska
Z	Kodiak, Alaska

Table V. North Pacific Loran-C Chain Transmitter Locations. [Ref. 1]

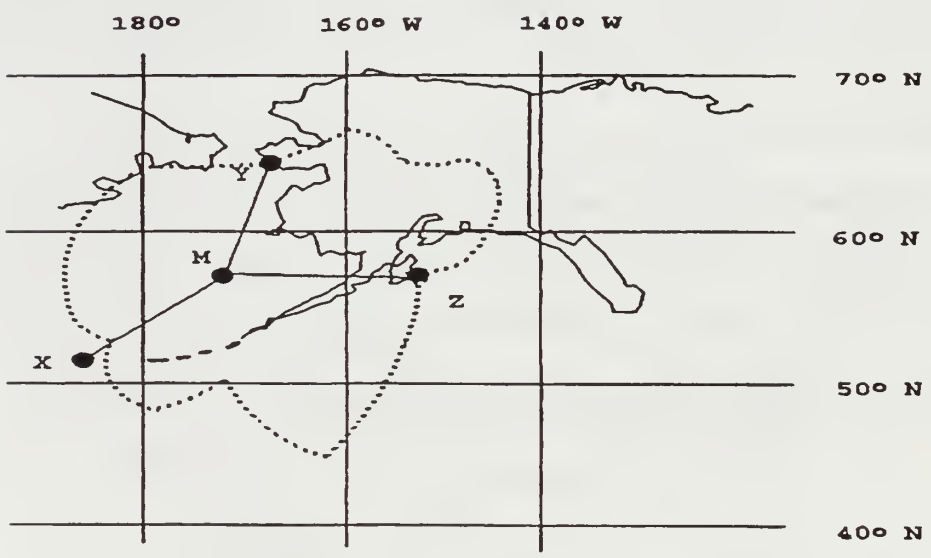


Figure 11. North Pacific Loran-C Chain. Transmitter Stations: M - St. Paul, AK; X - Attu, AK; Y - Port Clarence, AK; Z - Kodiak, AK. [Ref. 1]

2. Limitations

There were several limitations in the data sets used in this work. The most significant of these was that the data sets did not contain 24 hours worth of data

points. The initial data sets from Malone, Florida were missing anywhere from 65 minutes to as much as 164 minutes of data points. This precluded an exact correlation with the CALOC-generated strip charts and forced a different approach for comparison. After isolating and correcting the cause of the missing data points, the data sets from Middletown, California, were only missing between 7 and 17 minutes worth of data. Due to this limitation, each data set was used to generate strip charts that included only the data points.

A second limitation was that the data from each location (Malone, Florida, Middletown, California, and Kodiak, Alaska) covered no more than two days during the same time of the year. No comparison could be made to determine seasonal trends nor to verify that the data received was normal for each time of year. In order to perform a rigorous treatment of the data and to implement a controller, a significant amount of data would need to be collected during different seasons and under varying atmospheric conditions. Since this was not the case, additional testing on actual equipment under varying conditions is required prior to implementation of the proposed algorithm.

A third limitation was that the data sets included the effects of CALOC. Due to Loran-C's significant role in coastal navigation, there was no acceptable method to collect uncorrected data from an operational Loran-C chain; without CALOC's control input, the Loran-C chain would transmit an inaccurate signal during the time of data colletion. This required deriving a method for removing as much of CALOC's influence as possible so that the new contoller would have raw data that only included the inherent noise due to the propagation path, man-made interference, atmospheric noise, etc.

3. Data Preprocessing

There was little that could be done to mitigate the limitations of incomplete data sets and limited seasonal variation; however, it was believed that the effects of CALOC on the data set could be removed. The data sets, once plotted, revealed a

significant delay between the time CALOC ordered an LPA and the time that its full effect was detected at the monitoring station. Thus, the first step in removing the effect of CALOC on the data was to determine the average time of this processing delay. Data preprocessing for each data set was accomplished using the Matlab code contained in Section 5 of Appendix D. After loading the data set, the preprocessing involves manually determining the timing of LPAs within the data set and then removing them using a time-phased correction described below.

a. Processing Delay

Based on the strip charts generated by CALOC that were provided with the data sets, it appeared that the time delay experienced between CALOC ordering a correction and its effect being detected at the monitor station was relatively constant. As a first order approximation, it was assumed that this delay was less than the 7.5 minute sampling interval presently used in CALOC. However, upon further investigation, it was found that the average time delay was on the order of 56 data points, or over nine minutes. Identification of the source of this time delay is beyond the scope of this work.

To determine the magnitude of the time delay, a moving average over 45 data points, equating to CALOC's 7.5 minutes average interval, was taken of each data set and plotted. The time delay was displayed in the graph by a vertical change in the time difference error (TDE) as may be seen in Figure 12. The time delay was determined by counting the number of 10-second samples between the indication in the data of an LPA and the indication that the pulse transmitter had shifted to the new transmission time.

b. Corrected Data

Based on 146 LPAs in the nine data sets, the average time delay was computed to be 556.4 seconds. In order to apply the correction to remove the LPAs due to CALOC, the time delay was rounded up to 560 seconds, which equated to 56 10-second data points. The LPAs were then removed by applying the inverse of the LPA uniformly over the 56 data points. The same data described above,

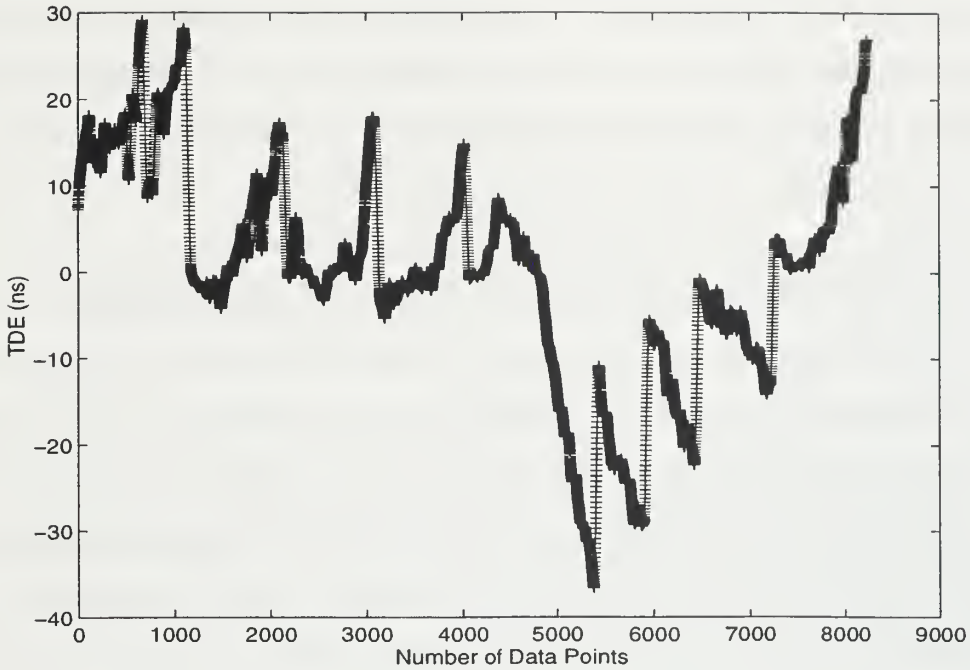


Figure 12. South-East United States Loran-C Station Yankee Time Difference Error recorded on 18 January 1997.

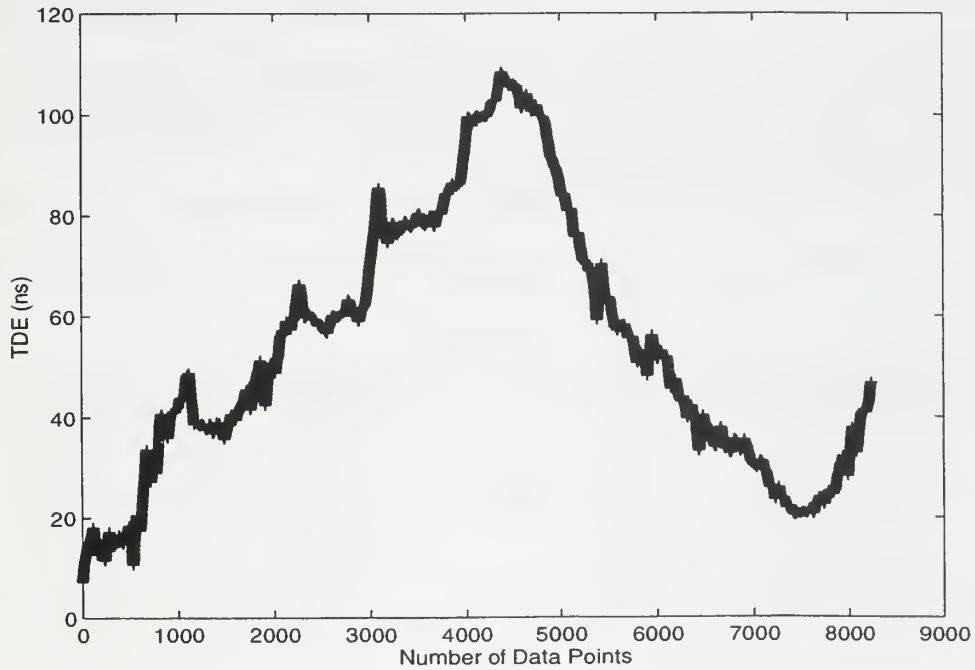


Figure 13. South-East United States Loran-C Station Yankee Time Difference Error recorded on 18 January 1997 with CALOC Local Phase Adjustments (LPAs) Removed.

after removing the LPAs, may be seen in Figure 13. The time delay displayed no dependence on the magnitude of the ordered LPA since it was consistent for both a ±20 ns LPA or a ±40 ns LPA. This indicated that the delay was due solely to the time required to apply the LPA to the pulse transmitter.

B. FINITE IMPULSE RESPONSE (FIR) FILTER

The recorded Loran-C data contains high frequency noise components that need to be removed. These are due primarily to noise sources in the wireless medium, but also include noise from the transmitter and receiver. Additional noise is injected in the data during the preprocessing. To remove these high frequency noise sources, a digital low-pass filter is required. There are two basic types of digital filters: finite impulse response (FIR) and infinite impulse response (IIR). An FIR filter was selected primarily due to its linear phase characteristic. Other benefits of the FIR filter are that it is simpler to design and, when windowed, is of finite duration.

There are several methods of FIR filter design that may be used, including frequency sampling and windowing. The windowing method truncates an infinite duration unit sample response, $h_d(n)$, to length $M-1$ by multiplying $h_d(n)$ by a rectangular window of length M, defined as:

$$w(n) = \begin{cases} 1, & n = 0, 1, \ldots, M-1 \\ 0, & \text{otherwise.} \end{cases} \tag{4.1}$$

A symmetric lowpass linear-phase FIR filter is described in the frequency domain as:

$$H_d(\omega) = \begin{cases} e^{-j\omega\frac{(M-1)}{2}}, & 0 \leq |\omega| \leq \omega_c \\ 0, & \text{otherwise.} \end{cases} \tag{4.2}$$

A windowed FIR filter of length M has a unit sample response

$$h(n) = \frac{\sin\omega_c(n-\frac{M-1}{2})}{\pi(n-\frac{M-1}{2})}, \qquad 0 \leq n \leq M-1, n \neq \frac{M-1}{2} \tag{4.3}$$

where ω_c is the cut-off frequency and M is the filter length. Using a rectangular window has one significant drawback; it causes relatively large sidelobes. To eliminate the sidelobes, a window function that gradually decays toward zero is desirable. [Ref. 10]

There are several windowing options available. Here we use the Hamming window, which is given as:

$$w(n) = 0.54 - 0.46\cos(\frac{2\pi n}{M-1}), \qquad 0 \leq n \leq M-1 \tag{4.4}$$

where M is the window length or filter length. The Hamming window has significantly lower sidelobes compared to the rectangular window. However, for the same length filter, the width of the main lobe is wider which leads to a wider transition region in the FIR filter response. [Ref. 10] The filter used in this application is an FIR filter with a Hamming window of length 32 and a cut-off frequency of $\pi/45$. The cut-off frequency was selected based on the removal of all high frequency components beyond the 45 data points used in each average.

The FIR filter is generated using the function *firham.m* (see Section 2, Appendix D). The inputs to the function are the window length, M, and the cut-off frequency, ω_c; the outputs are the filter coefficients. For $M = 32$, $\omega_c = \pi/45$, and using a Hamming window, the coefficients of the FIR filter are listed in Table VI.

h(0) = h(31)	0.0040
h(1) = h(30)	0.0046
h(2) = h(29)	0.0062
h(3) = h(28)	0.0088
h(4) = h(27)	0.0123
h(5) = h(26)	0.0166
h(6) = h(25)	0.0217
h(7) = h(24)	0.0272

h(8) = h(23)	0.0330
h(9) = h(22)	0.0389
h(10) = h(21)	0.0445
h(11) = h(20)	0.0496
h(12) = h(19)	0.0540
h(13) = h(18)	0.0575
h(14) = h(17)	0.0600
h(15) = h(16)	0.0612

Table VI. FIR Filter Coefficients.

The response of the FIR filter prior to windowing with the Hamming window is depicted in Figure 14. The peak of the first sidelobe is approximately -20 dB below the passband peak, and the phase response is linear over the passband. After windowing, the filter response, shown in Figure 15, has a peak sidelobe level of approximately -45 dB while maintaining the linear phase. As mentioned above, the transition band has increased when using the Hamming window compared to the rectangular window.

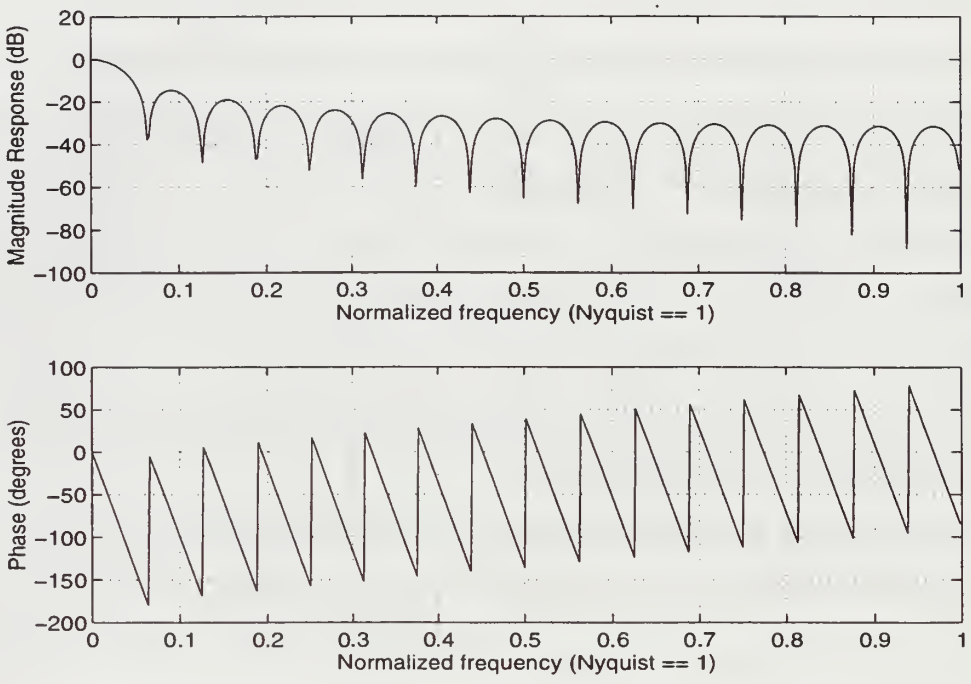


Figure 14. Magnitude and Phase Response for Finite Impulse Response (FIR) Filter (Length = 32, Cut-off Frequency = $\pi/45$) with no Windowing (i.e., rectangular window).

C. PROPORTIONAL-INTEGRAL-DERIVATIVE (PID) CONTROLLER

This section presents a PID controller. The process began with exploring a model of the existing control system and implementing that model. Based on the model, suitable weights were derived for the controller, and then the algorithm was tested using all the data sets.

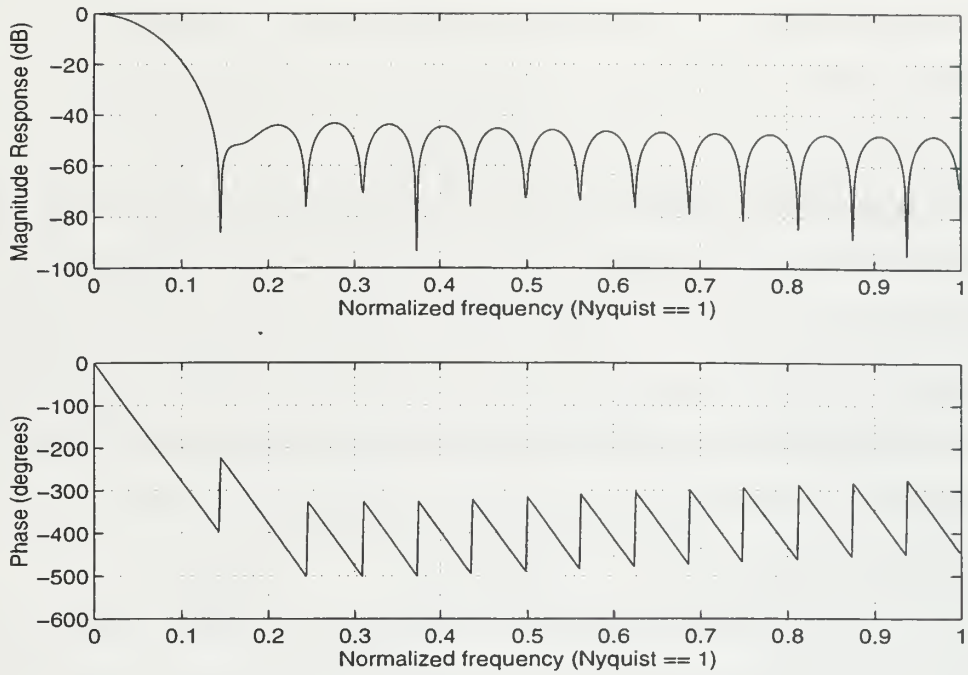


Figure 15. Magnitude and Phase Response for Finite Impulse Response (FIR) Filter (Length = 32, Cut-off Frequency = $\pi/45$) with Hamming Window.

1. Theory

The PID controller is widely used in industrial processes to control systems that cannot be modeled as lumped and linear time-invariant. It is essentially three controllers in one: a proportional controller, an integral controller and a derivative controller. [Ref. 11]

The transfer function of the proportional controller is a gain, k_p. For a given controller input, $e(t)$, the output will be:

$$u(t) = k_p e(t). \tag{4.5}$$

For a stable plant, the feedback system will remain stable for a range of k_p. However, as k_p increases, the unit-step response may become faster and the feedback system becomes unstable. One significant limitation to the proportional controller is that for the same unit-step reference input, the steady-state plant output will be different for different values of k_p requiring the plant set point to be manually reset. [Ref. 11]

For the same input, the integral controller output is:

$$u(t) = k_i \int_0^t e(\tau)d\tau \tag{4.6}$$

where k_i is the integral constant. This leads to a transfer function for the integral controller of $\frac{k_i}{s}$. For a stable feedback system, the steady-state error due to any step-reference input is zero for all k_i. This means that there is no need to manually reset the set point when k_i is changed. For this reason, integral control is also called reset control with k_i being the reset rate. Though the requirement to reset the set point has been eliminated, the integral controller makes stabilizing the feedback system more difficult. [Ref. 11]

The derivative controller maintains control of a system by using the rate of change of the error signal as its input. The output of the derivative controller is:

$$u(t) = k_d \frac{de(t)}{dt} \tag{4.7}$$

where $e(t)$ is the input to the controller and k_d is the derivative constant. The derivative controller by itself is rarely used in feedback control systems because if the error signal changes slowly or is constant, the controller output would be minimal or zero allowing a potentially large error to remain. [Ref. 11]

A PID controller simultaneously takes advantage of all three controllers described above. The three parameters of the controller, k_p, k_i and k_d, are tuned to achieve the desired system response. One method of tuning is by trial-and-error, initializing each parameter to some constant value and varying all three parameters to achieve a satisfactory system output. Exact values of these parameters could not be obtained due to lack of an accurate model for the entire Loran-C control subsystem. Because of this, a trial-and-error approach was used to obtain the solution described in this chapter. [Ref. 11]

2. Model

The most important step in the process of developing a control algorithm is constructing an accurate model. The system model is a mathematical representation of the physical system. It may be used to evaluate the performance of the physical system to different input stimuli when the system is unavailable for operational testing, as is the case for the Loran-C system. The model developed for the existing control system, shown in Figure 16, contains four major sections, each modelled as separate transfer functions: CALOC, the transmitter, the medium and the monitoring station.

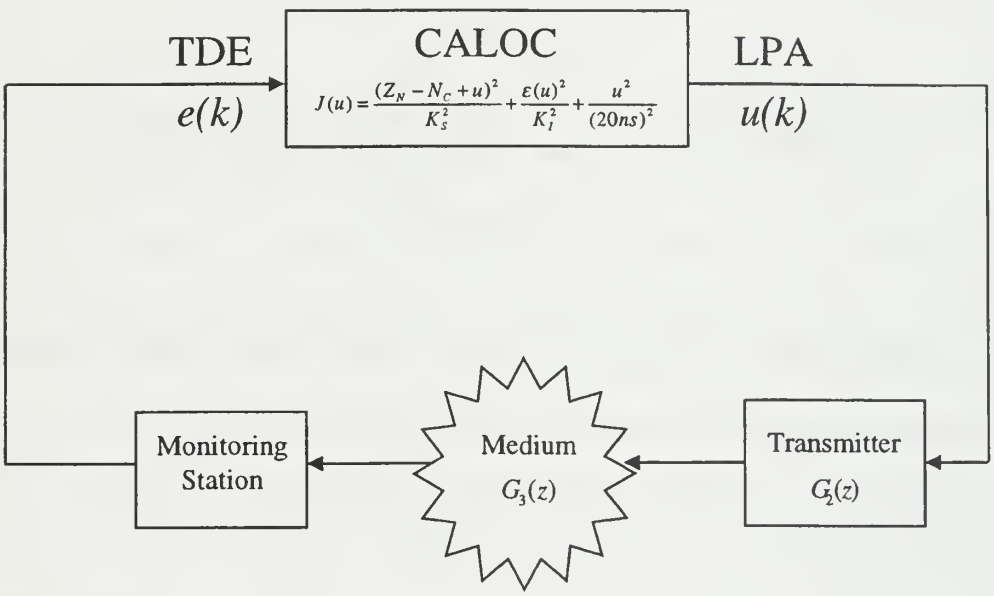


Figure 16. Loran-C Control System Model.

Knowledge of the transfer functions for the transmitter, medium and the monitoring station is not required based on the method of data preprocessing discussed earlier. The preprocessing only removed the effect of CALOC while leaving intact all noise components and the dynamic behavior of the transmitter, the medium, and the monitoring station equipment. This significantly simplified the process of modelling the new control system. All that was required was to substitute the new control

transfer function into the block that contained the CALOC transfer function. The new model is depicted in Figure 17.

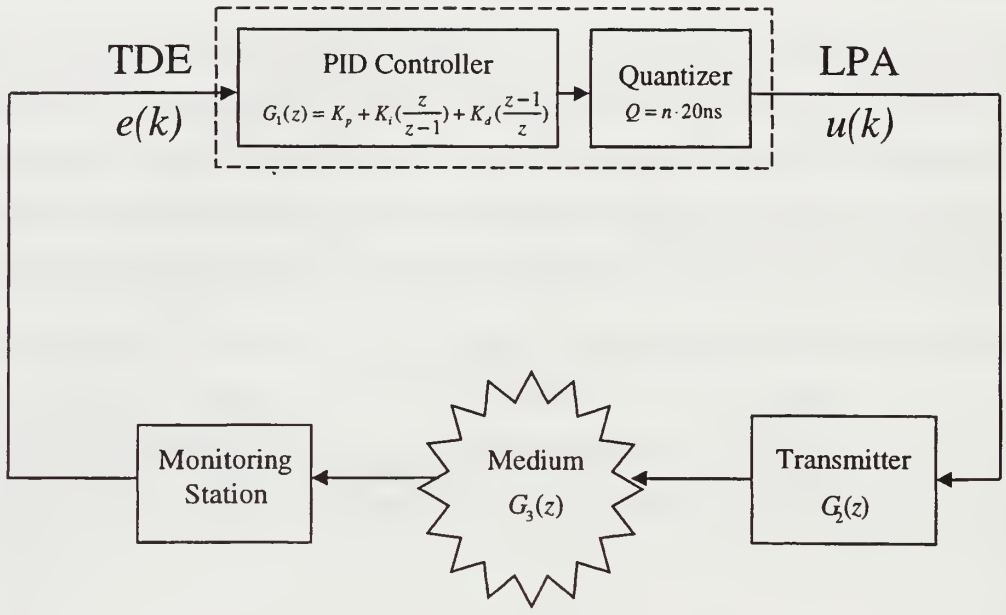


Figure 17. Representative Model of the Loran-C Control System with PID Controller.

Some modifications to the model were required to support simulation of the new control system. A method to input local phase adjustments (LPAs) in the data was developed to simulate the functionality of the new control algorithm within the Loran-C control system. As depicted in Figure 18, the simlation model sums the generated LPA with the future time difference errors (TDEs) from the recorded data. The LPA process was modelled as an integrator and time delay with a transfer function of:

$$G_L(z) = \frac{z}{z-1} z^{-1}. \tag{4.8}$$

The LPA is a timing adjustment to the Loran-C station pulse transmitter for all future transmissions which leads to modelling the LPA function as an integrator.

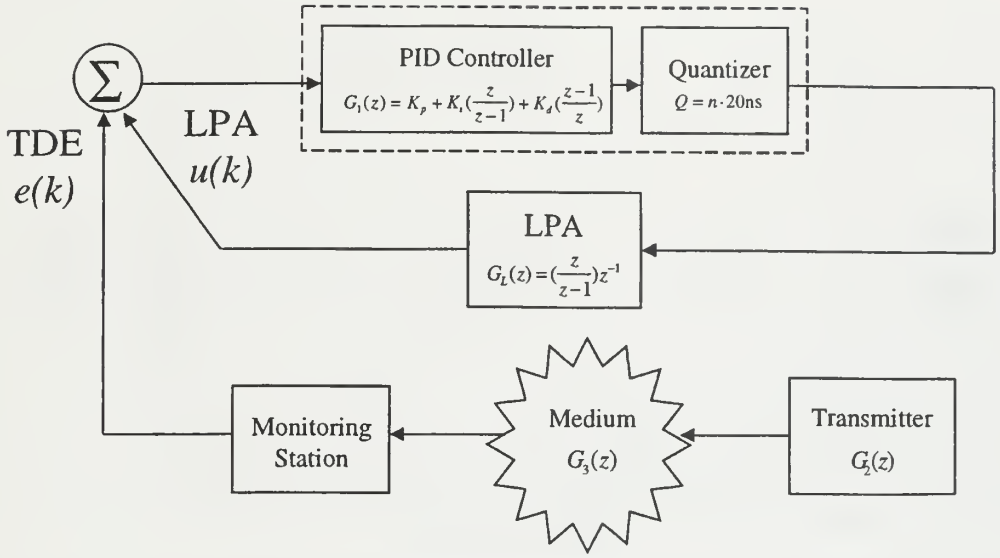


Figure 18. Simulation Model of the Loran-C Control System.

The single time delay was based on the fact that the effect of the LPA was detected at the monitoring station in about one 7.5-minute control interval.

The algorithm for the PID controller is the classical model shown in Figure 19. The input to the controller is the TDE at time k and the output is the continuous time LPA recommendation for time k. The proportional gain, K_p, is applied to the TDE directly. The integral of the TDE is the cumulative TDE and is controlled using the integral gain, K_i. The error rate of change is calculated by taking the difference between $e(k)$ and $e(k - 1)$. The derivative gain, K_d, is applied to this difference. Summing all three contributions of the PID controller yields the LPA output that is then provided as input to the quantizer, which yields discretized values, $u(k)$.

3. Quantization

Quantization is most commonly used in analog-to-digital (A/D) conversion to convert a continuous time signal into a set of discrete values. Here, quantization is used to transform the LPA output of the controller into quanta of 20 ns, over a range of ± 180 ns, for use in controlling pulse transmission timing. There are two basic types

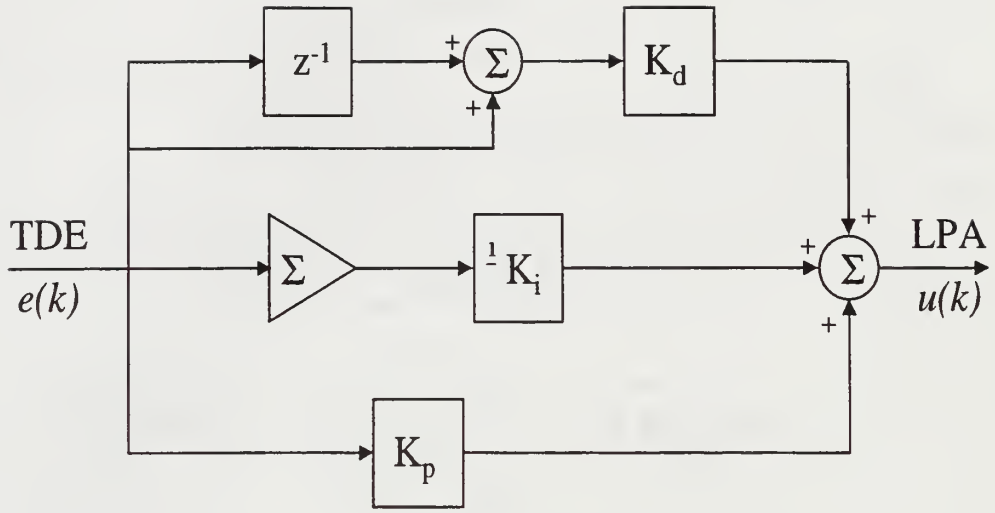


Figure 19. PID Controller without Quantization.

of quantizers: mid-rise and mid-tread. The mid-tread quantizer is generally preferred over the mid-rise because the quantization noise has zero mean. In this application, the mid-tread is used as it did not introduce a bias to the signal. [Ref. 10]

Quantization does not come without some cost. The process of quanitization introduces an error that cannot be removed. Figure 20 depicts the uniform, also known as linear, quantizer designed for this controller. The step size is a uniform value of 20 ns, which yields a maximum quantization error of 10 ns, or half the step size, throughout the dynamic range of the quantizer. The quantization error for each sample may be modeled as a zero-mean, uniform random variable over the interval ± 10 ns.

D. PID RESULTS

The PID control algorithm was implemented using the function *pid.m* (see Section 3, Appendix D). The inputs to the function are the data set name, initial condition for the cumulative TDE, and title for the plot. The plot title is generated by the function *parse_title.m* (see Section 7, Appendix D).

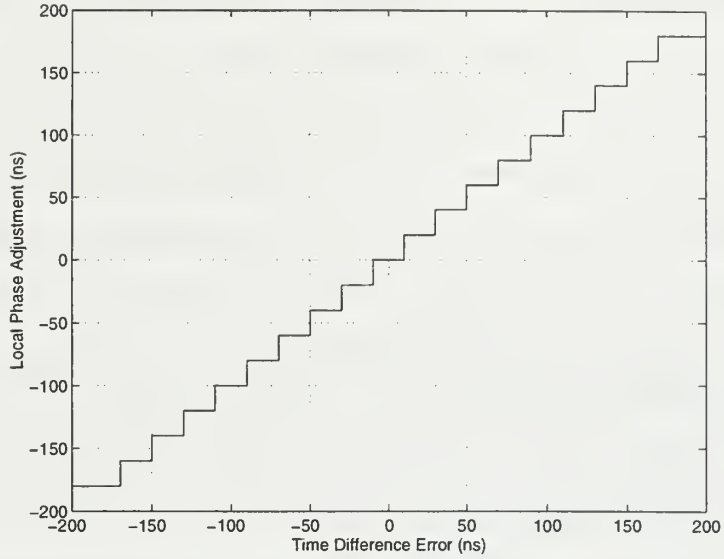


Figure 20. PID Controller Output Quantizer for Generating Local Phase Adjustments.

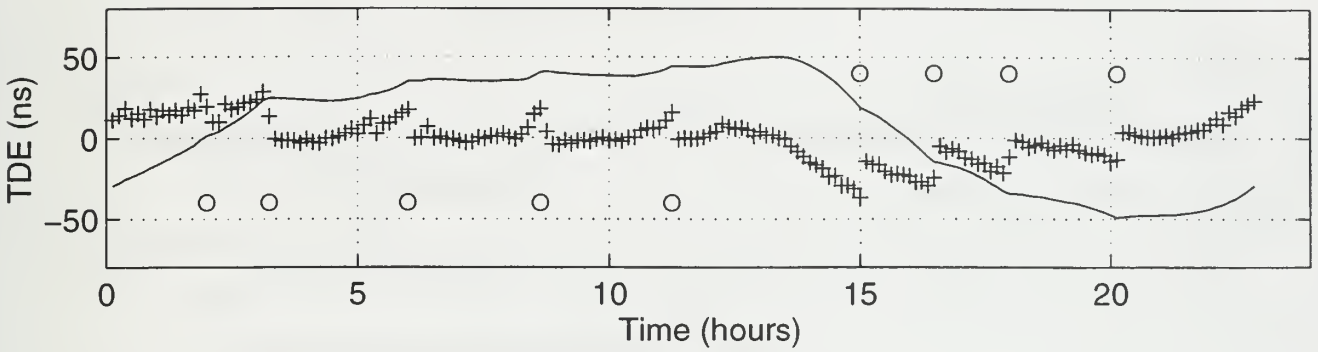
The PID control algorithm proved to be a definite improvement over the curve-fit algorithm in the existing Loran-C controller. In every case, both the current time difference error (TDE) and the cumulative TDE are maintained in closer control. In some cases, the PID controller was even able to accomplish this with fewer local phase adjustments (LPAs) than CALOC. However, in eight cases out of 29 there are a large number of LPAs being ordered which is counter to one of the USCG control policies: to minimize the number of LPAs. This excessive chatter of the controller could be minimized with further operational testing using a greater number of data sets.

The results are presented for two Loran-C stations: SEUS Station Yankee and SEUS Station Zulu. The results of the other 27 data sets are included in Appendix B. SEUS Station Yankee cumulative TDE was initialized at -31 ns based on the actual value found on the CALOC strip chart. As is shown in Figure 21, CALOC maintained the current TDE within a range of -37 ns to 28 ns while the cumulative TDE has a range of ± 50 ns. The PID controller showed a 14% improvement in dynamic range in both current TDE, with a range of -30 ns to 26 ns, and cumulative TDE, with a range of -40 ns to 46 ns.

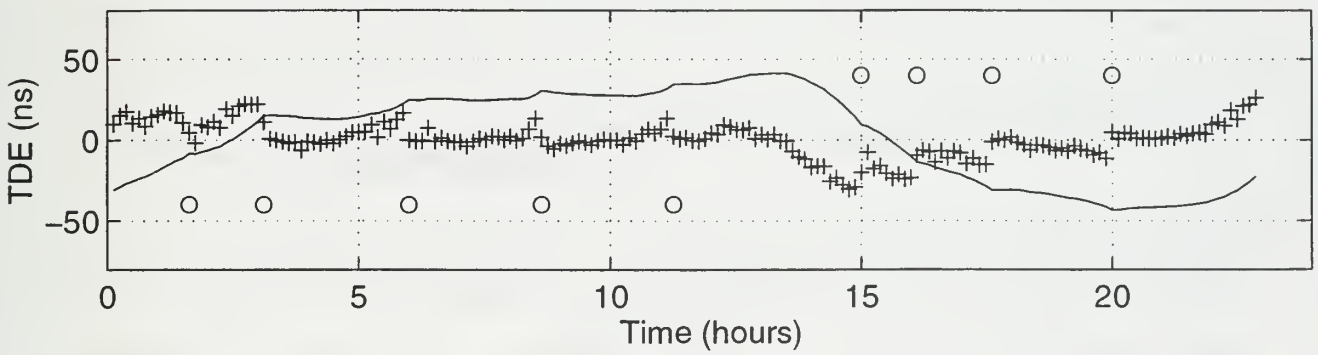
The results for SEUS Station Zulu were even better. The cumulative TDE was initialized at -77 ns. As seen in Figure 22, CALOC was able to maintain the current TDE between -38 ns and 32 ns and the cumulative TDE between -71 ns and 38 ns. The PID controller was able to provide a significantly smaller dynamic range of cumulative TDE for Station Zulu. For the PID controller, the current TDE had a range of ± 31 ns, an 11% improvement, and the cumulative TDE was maintained between -53 ns and 30 ns for a 24% improvement.

The results of the PID controller on the 29 data sets indicate that it is a viable option to CALOC. There was improved control of the TDE in every case though some data sets required a greater number of LPAs than the same data under CALOC. However, in the case of the data from the United States West Coast (USWC) Chain and the North Central United States (NOCUS) Chain, the chains are rarely in automatic control, which allows CALOC to function fully with no human intervention. Thus, the direct comparison with CALOC on those chains is not as conclusive.

Based on the overall results and the knowledge of the ever-changing medium in which Loran-C operates, the control algorithm needs to be data-dependent to better respond to this high noise environment. A PID controller could provide the needed adaptability; however, there are other controllers that are better suited. One of these is the Kalman filter, which is developed in the next chapter.

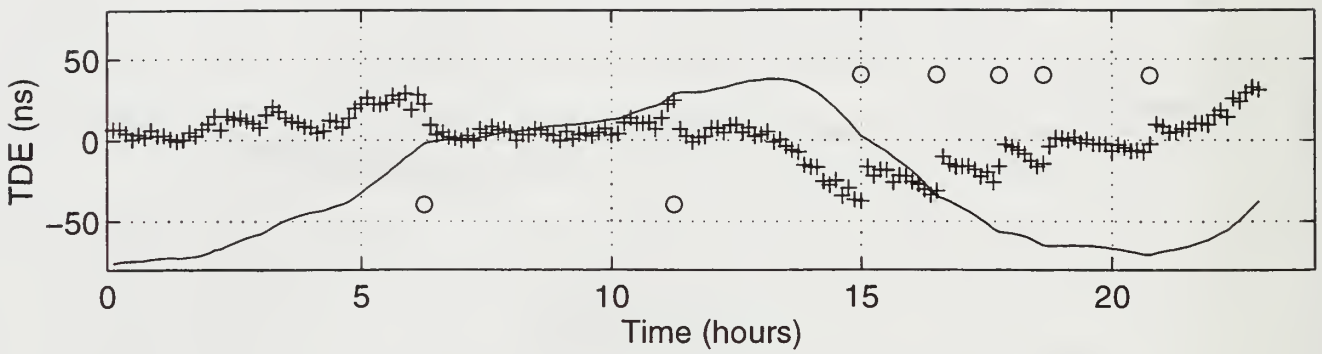


(a) CALOC

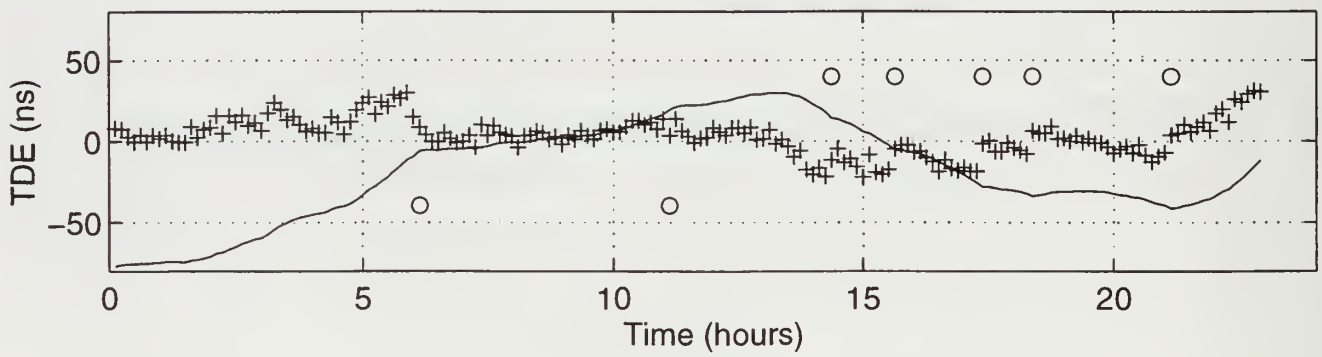


(b) PID Controller

Figure 21. Strip Chart for South East United States (SEUS) Loran-C Chain Station Yankee recorded on 18 January 1997: o Linear Phase Adjustment (LPA) (magnitude times 2); + Time Difference Error (TDE); — Cumulative TDE.



(a) CALOC



(b) PID Controller

Figure 22. Strip Chart for South East United States (SEUS) Loran-C Chain Station Zulu recorded on 18 January 1997: o Linear Phase Adjustment (LPA) (magnitude times 2); + Time Difference Error (TDE); — Cumulative TDE.

V. DATA-DEPENDENT CONTROL ALGORITHM

The PID controller was an initial step in determining whether significant improvement over CALOC was possible. As was proved in the preceding chapter, there is significant room for improvement. This chapter will demonstrate the effects of a more robust controller, the Kalman filter. One of the advantages of the Kalman filter is its ability to adapt to changes in the operational environment. The discussion presented here is introductory, and the results reported are preliminary.

A. DATA-DEPENDENT SOLUTION

The primary reason for developing a data-dependent solution for the Loran-C control problem is the operating environment of the signal. The signal environment is a wireless media that is subject to these sources of interference, both natural and man-made. These sources of interference include atmospheric changes, terrain and electromagnetic intereference.

1. Atmospheric Effects

The sources of atmospheric effect on the Loran-C signal are primarily due to temperature changes and seasonal weather effects. The effect of temperature changes on signal propagation has no reasonable theoretical explanation other than its effect on the refractive index of the media. Specifically, the Loran-C signal exhibits a diurnal variation that can be associated with temperature changes. [Ref. 12] One source of these temperature effects is a temperature inversion and associated humidity change which may produce highly variable refractivity profiles. This process produces an evaporation duct in which radio waves may propagate well beyond the horizon. Overland propagation refraction effects may be seen in at least two situations: one is the evaporation of water on the ground after raining which produces an over-land

evaporation duct, and the second is the radiational cooling over deserts and snow-covered terrain on clear nights which yields anomalous propagation. [Ref. 13]

The seasonal weather effects on the Loran-C signal are caused by the passage of cold and warm fronts predominate at specific times of the year. The interpretation of the correlation between the time difference and total refractivity is that the passage of cold and warm fronts produce transient decreases in the propagation time delay. The cold front decreases occur after the front has passed while the decreases associated with the warm front occur before frontal passage. [Ref. 12]

The conclusion that may be drawn from these effects is that the Loran-C signal is being influenced by changes in the refractive index well above the earth's surface. A more complete understanding of these atmospheric effects is required in order to make predictions. In the case of the Loran-C system, accurate predictions correspond directly with greater position accuracies. [Ref. 12]

2. Terrain

The signal effects of terrain on the Loran-C signal are due to diffraction phenomena, which are exhibited in both diffuse scattering and coherent scattering. These phenomena may be broken down into two categories. The first is specific irregularities in the terrain, such as mountain ridges, which are often modeled as knife-edges or, in some cases, cylinders. Secondly, diffraction is produced by the curvature of the smooth spherical surface of the earth which applies to propagation over the ocean or in the plains. At low frequencies, diffraction by hills or ridges alters the field strength in a significant region around the line of sight which means that more than one diffracting feature on the terrain may be contributing to the propagation losses. This is a challenging issue that has not yet been successfully modeled for long ray-paths over varying terrains. [Ref. 13]

3. Electromagnetic Interference

The sources for electromagnetic interference for the Loran-C system are essentially of two types: fixed radiators and mobile, or random, radiators. The fixed radiators are those that continuously transmit at a constant frequency or transmit at consistent times of the day while the mobile radiators randomly transmit their signal and maintain no pattern to their transmission. Each master-secondary baseline will have a different set of known radiators that may be relatively easily modeled. However, determining a suitable model for the mobile radiator may be quite involved and location dependent.

Two fixed radiators that are significant contributors to electromagnetic interference are overhead power lines and radio towers. The power lines are low power signals that exist within the Loran-C frequency band. This has become of particular interest since the expansion of the Loran-C system across the North American continent. The signal amplitudes vary as a function of the square of the distance from the power line, so their effect is only critical near to the power line, on the order of one mile. However, because so many power lines exist, using notch filtering to remove the effects is not feasible. Radio towers along the ray-path also contribute to the electromagnetic interference seen by the Loran-C signal. One example of this is the frequency shift keying found at naval communications stations. These, however, unlike the power lines, may be divided using superposition and treated as either deterministic or random noise when considering their effects on the time of arrival. [Ref. 14]

One example of the mobile, or random, radiator that causes electromagnetic interference with the Loran-C signal is the cathode ray tube (CRT). The raster scan displays on computers and test instruments cause interference due to the harmonics of the horizontal scan frequency. These frequencies are extremely stable and may be isolated once the device is identified. [Ref. 14] There are other sources of random electromagnetic interference that would also need to be accounted for. One prominent

example is the ham radio operator that may be infrequently transmitting near the operating frequency of Loran-C. The mobile radiator presents a unique challenge in that their signal, in the aggregate, is a non-deterministic, random process.

B. KALMAN FILTER

Kalman filtering provides several advantages over the PID algorithm in solving the Loran-C control problem. Its mathematical formulation is described in state-space form, and its solution is computed recursively. Thus, each updated state estimate is computed only from the previous estimate and the new input data which requires only that the previous state estimate be stored. The Kalman filter is also more efficient than computing the next state estimate using the entire past observed data at each step in the process. [Ref. 15]

1. Model

In the general case, the Kalman filtering problem is stated in terms of two vectors of random variables: an M-dimensional parameter $\mathbf{x}(k)$ which denotes the state of a discrete-time, linear, dynamic system and $\mathbf{y}(k)$ which denotes the observed data of the system. The system model may then be fully described by two equations: a process equation and a measurement equation. [Ref. 15]

The process equation is:

$$\mathbf{x}(k+1) = \mathbf{A} \cdot \mathbf{x}(k) + \mathbf{v}(k) \tag{5.1}$$

where $\mathbf{A}$ is an $M \times M$ state transition matrix that relates the system state at times k and $k+1$. The $M \times 1$ vector $\mathbf{v}(k)$ represents process noise and is modeled as a zero-mean, white-noise process, ideally Gaussian. [Ref. 15]

The measurement equation, which describes the observation vector, is:

$$\mathbf{z}(k) = \mathbf{C} \cdot \mathbf{x}(k) + \mathbf{\Delta w}(k) \tag{5.2}$$

where $\mathbf{C}$ is an $N \times M$ measurement matrix. The $N \times 1$ vector $\Delta\mathbf{w}(k)$ represents measurement, or sensor, noise and is modeled as another zero-mean, white-noise process, ideally Gaussian. [Ref. 15]

The Kalman filter attempts to use the past observed data values, consisting of $\mathbf{y}(0), \ldots, \mathbf{y}(k)$, and the model to predict the minimum mean-square estimate of the new state, $\mathbf{x}(k+1)$. Thus, the Kalman filter is attempting to solve the equation:

$$\mathbf{x}(k+1) = \mathbf{A} \cdot \mathbf{x}(k) + \mathbf{L}(k) \cdot [\mathbf{z}(k) - \mathbf{C} \cdot \mathbf{x}(k)] \tag{5.3}$$

where $\mathbf{z}(k)$ - $\mathbf{C} \cdot \mathbf{x}(k)$ is the correction to be applied and $\mathbf{L}(k)$ is determined in order to minimize $\Sigma \parallel \tilde{\mathbf{x}}(k) \parallel^2$ where the error, $\tilde{\mathbf{x}}(k) = \mathbf{x}(k)$ - $\hat{\mathbf{x}}(k)$.

In tailoring the general model to the specific case of the Loran-C control problem, depicted in Figure 23, several adaptations were made. In Equation 5.1, the first term becomes $\mathbf{x}(k) + \Delta\Phi(k)$ yielding a new process equation of

$$\mathbf{x}(k+1) = \mathbf{x}(k) + \Delta\Phi(k) + \mathbf{v}(k) \tag{5.4}$$

where x(k) is the estimate of the current TDE and $\Delta\Phi(k)$ is the input LPA. Similarly, the first term in Equation 5.2 becomes $\mathbf{z}(k) + \mathbf{x}(k)$ yielding a new measurement equation of

$$\mathbf{z}(k+1) = \mathbf{z}(k) + \mathbf{x}(k) + \Delta\mathbf{w}(k) \tag{5.5}$$

where $\mathbf{z}(k)$ is the cumulative TDE.

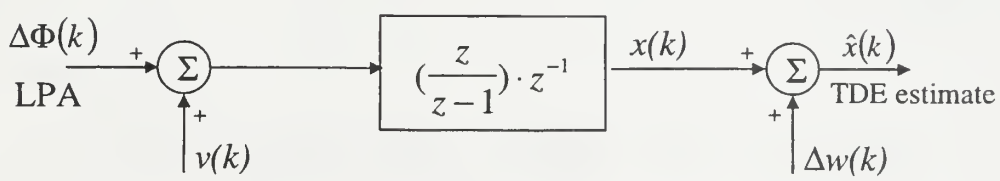


Figure 23. Control System Process Model.

As discussed in Chapter III, the long-term fluctuations of 12-24 hours were of lesser consequence since the control of the time difference (TD) would be over a much shorter period. A suitable model for the short-term fluctuations in the time difference error (TDE) is a random walk with additive white Gaussian noise as depicted in Figure 24. The model is characterized by the equation:

$$\mathbf{w}(k) = \mathbf{w_m}(k) + \mathbf{\Delta w}(k) \tag{5.6}$$

where

$$\mathbf{w_m}(k) = \mathbf{w_m}(k-1) + \mathbf{v}(k-1) \tag{5.7}$$

with $\mathbf{w}(k)$ and $\mathbf{v}(k)$ being zero mean, uncorrelated, white Gaussian noise.

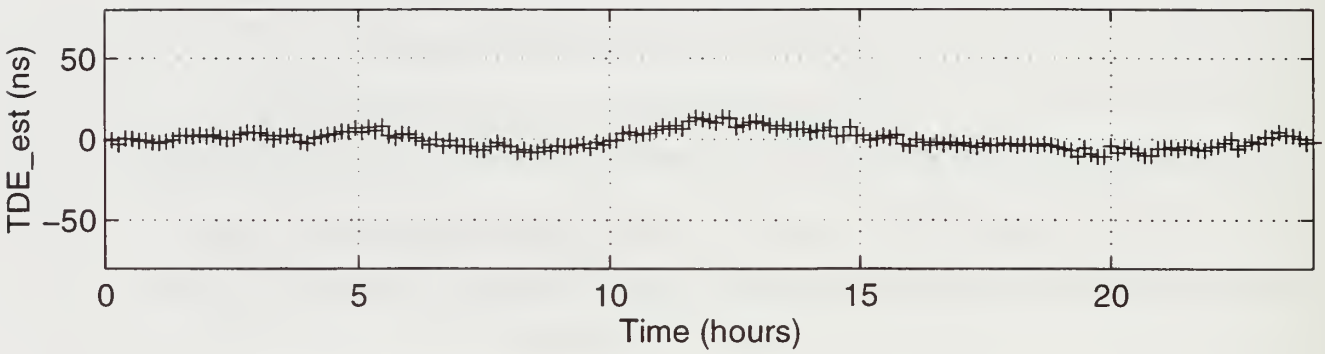


Figure 24. Random Walk with Additive White Gaussian Noise

2. Optimal Controller

Based on the model described above, the Kalman filter estimates the whole state as defined by:

$$\hat{\mathbf{x}}(k+1) = \hat{\mathbf{x}}(k) + \mathbf{\Delta\Phi}(k) + \frac{\mathbf{p}(k)}{\mathbf{R}+\mathbf{p}(k)}(\mathbf{\Phi_e}(k) - \hat{\mathbf{x}}(k)) \tag{5.8}$$

$$\mathbf{p}(k+1) = \mathbf{p}(k) + \mathbf{Q} - \frac{\mathbf{p}(k)^2}{\mathbf{R}+\mathbf{p}(k)} \tag{5.9}$$

$$\hat{\mathbf{z}}(k+1) = \hat{\mathbf{z}}(k) + \hat{\mathbf{x}}(k) \tag{5.10}$$

where $\hat{\mathbf{x}}(k)$ is the TDE estimate, and $\hat{\mathbf{z}}(k)$ is the estimate of the cumulative TDE. The constants $\mathbf{Q}$ and $\mathbf{R}$ are defined as

$$\mathbf{Q} = \text{cov}(\mathbf{v}(k)) \tag{5.11}$$

$$\mathbf{R} = \text{cov}(\Delta\mathbf{w}(k)). \tag{5.12}$$

In order to devise an optimal controller, a control law must be defined. As discussed in the previous chapter, the USCG has established Loran-C control procedures concerning the allowed range of time difference and minimizing the number of LPAs. The metric that is used to measure the performance of the controller is a cost function, defined as:

$$\mathbf{J}(k) = \sum_{k=0}^{\infty} \{\sigma_1 |\hat{\mathbf{x}}(k)|^2 + \sigma_2 |\hat{\mathbf{z}}(k)|^2 + |\Delta\mathbf{\Phi}(k)|^2\} \tag{5.13}$$

where $\hat{\mathbf{x}}(k)$ is the estimate of the current TDE, $\hat{\mathbf{z}}(k)$ is the estimate of the cumulative TDE, and $\Delta\mathbf{\Phi}(k)$ is the LPA. The cost function is similar to the one defined for CALOC in that it applies a cost to a large TDE, a large cumulative TDE, and a large LPA. Control of the process is defined by

$$\Delta\mathbf{\Phi}(k) = -\mathbf{L}(k) \begin{bmatrix} \hat{\mathbf{x}}(k) \\ \hat{\mathbf{z}}(k) \end{bmatrix} \tag{5.14}$$

where $\mathbf{L}(k)$ is determined recursively using the Matlab function *dlqr*, which solves the discrete-time linear-quadratic regulator problem and the associated Riccati equations.

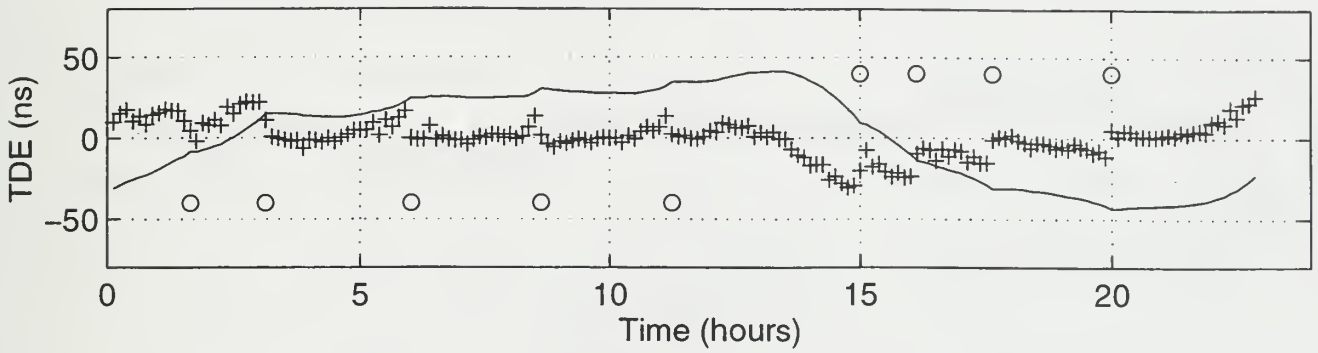
C. RESULTS

The Kalman filter was implemented using the function *kalman.m* (see Section 4, Appendix D). The inputs to the function are the data set name, initial condition for the cumulative TDE, and title for the plot. The plot title is generated by the function *parse_title.m* (see Section 7, Appendix D).

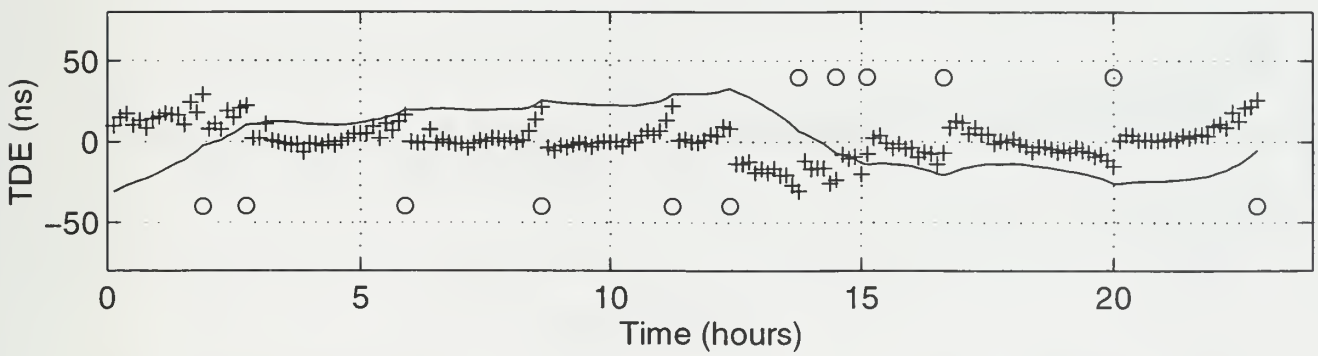
As in the case of the PID controller, the Kalman filter improved the control of the time difference error (TDE) over the existing Loran-C controller. It was also an improvement over the PID controller and, in most cases, inserted fewer local phase adjustments (LPAs). The results detailed below compare the PID control algorithm to the Kalman filter as the PID has already been shown to be superior to the existing controller. The same two Loran-C stations used in the preceeding chapter are used here to illustrate the improvement of the Kalman filter. The results of the other 27 data sets are included in Appendix C.

The initial cumulative TDE values for SEUS Stations Yankee and Zulu are the same as for the PID controller: -31 ns and -77 ns, respectively. As may be seen in Figure 25, the Kalman filter essentially maintained the same range for current TDE while improving control of the cumulative TDE. The range for the current TDE for the Kalman filter was -30 ns to 29 ns as compared to a range of -30 ns to 26 ns for the PID controller. However, the Kalman filter showed a 31% improvement in cumulative TDE with a range of -26 ns to 33 ns compared to the PID controller with a range of -40 ns to 46 ns.

The improvement of the results for SEUS Station Zulu were more pronounced. Similar to Station Yankee, Station Zulu maintained the current TDE within a similar range: -31 ns to 36 ns as compared to $\pm$31 ns for the PID controller. As Figure 26 illustrates, the range of cumulative TDE for the Kalman filter was -33 ns to 32 ns as compared to -53 ns to 30 ns for the PID controller, a 34% improvement.

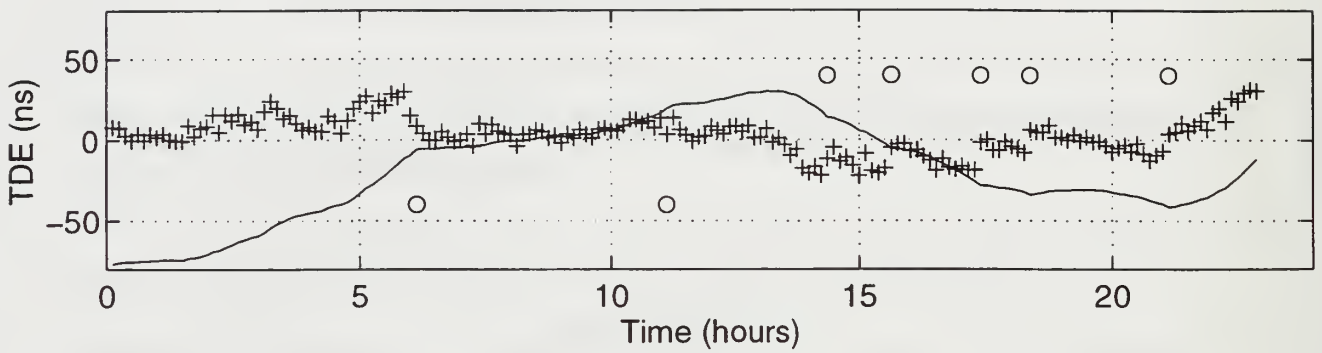


(a) PID Controller

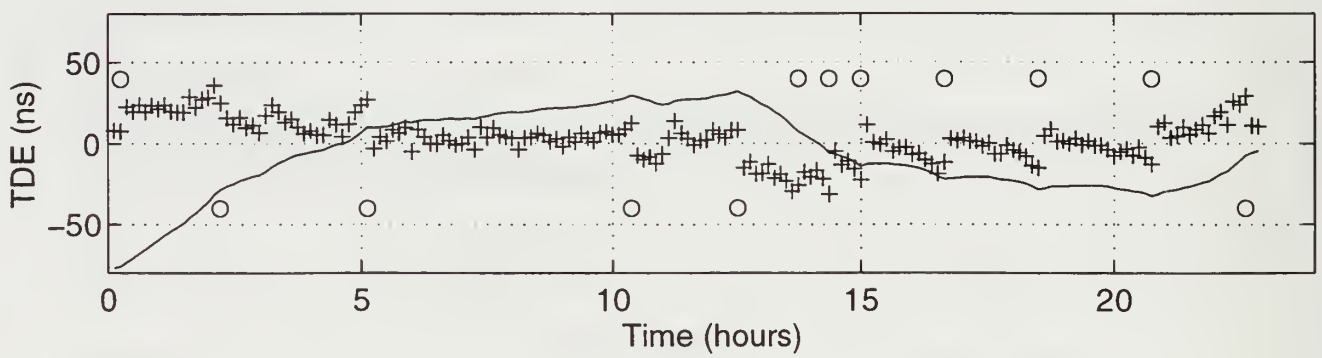


(b) Kalman Filter

Figure 25. Strip Chart for South East United States (SEUS) Loran-C Chain Station Yankee recorded on 18 January 1997: o Linear Phase Adjustment (LPA) (magnitude times 2); + Time Difference Error (TDE); — Cumulative TDE.



(a) PID Controller



(b) Kalman Filter

Figure 26. Strip Chart for South East United States (SEUS) Loran-C Chain Station Zulu recorded on 18 January 1997: o Linear Phase Adjustment (LPA) (magnitude times 2); + Time Difference Error (TDE); — Cumulative TDE.

VI. CONCLUSIONS

Loran has a long history of providing accurate navigational information to both mariners and aviators and can have a bright future into the next century as a complementary radionavigation system to GPS. To ensure this future of Loran-C, its accuracy must be enhanced and operational costs reduced. One method of making this improvement is to upgrade the transmitters and receivers in each of the LORSTAs, but this would be extremely costly. An alternative is to improve the control algorithm. Modernizing the control system for Loran-C requires a control algorithm that not only maintains the time difference error of the transmitted pulse near zero, but also is responsive to changing atmospheric conditions. To this end, this thesis developed a model of the existing system and proposed two control algorithms that provided substantial improvement over the existing algorithm: one was the PID controller and the other was a Kalman filter.

A. SIGNIFICANT RESULTS

Modelling Loran-C's ever-changing operational environment is extremely difficult. It includes several pieces of hardware and a variety of noise sources that all have unique characteristics. There currently exists no mathematical model of the Loran-C control system that could be used to simulate its functionality. Due to this, only the control process of inserting Local Phase Adjustments (LPAs) was modelled leaving the remaining system effects in place. This simplified the design effort since actual data was available that included these effects.

However, working with actual system data presented unique challenges. It was not possible to have a Loran-C chain provide inaccurate navigational signals simply to record information for this thesis. The effects of the current controller, CALOC, were included with the data and had to be preprocessed. These effects were removed by applying a correction to the data over the average time that elapsed before the

impact of the LPA was felt. This process provided the raw data for testing the proposed control algorithms.

An understanding of Loran-C, and in particular CALOC, was required in order to develop a suitable control algorithm. The study of Loran-C focused on one of the four tasks of CALOC, time difference control. The four major components of this control effort are CALOC, the LORSTA transmitter, the wireless medium, and the monitoring station. The effects of the last three components remained after the removal of the CALOC LPAs, allowing for the development of an algorithm without having to model each component.

The PID control algorithm blends the strengths of three controllers into one: a proportional controller, an integral controller, and a derivative controller. This algorithm was developed using a basic model and tuned using a trial-and-error approach to provide the best control. The results indicate that this algorithm provides more accurate pulse transmission timing than CALOC. Once tuned, the PID controller provided substantial improvement over the existing algorithm in all cases, though in some cases the frequency of timing corrections was excessive.

The Kalman filter was explored as an alternative to the PID controller. The filter provides the capacity to be adaptive in a changing environment and is much more robust than the PID controller. This is an extremely important part of what is needed in the future control system for Loran-C in today's technological environment. The preliminary results of the Kalman filter showed an even greater improvement over the existing controller than the PID controller. Because the Kalman filter will need to be more finely tuned to be completely operational, the transition from CALOC to the Kalman filter will need an intermediate step, and the PID controller is the best choice for this step.

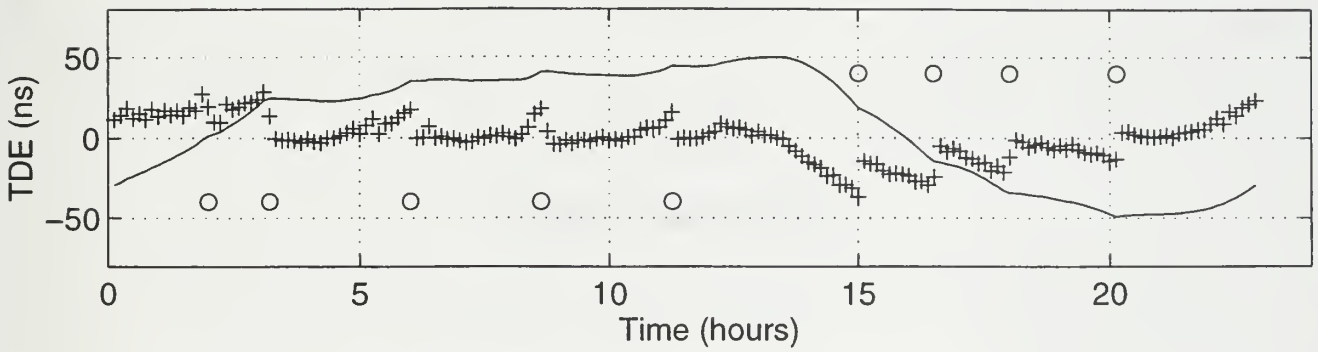
B. RECOMMENDATIONS FOR FURTHER STUDY

Further study of the Loran-C control system will be valuable in at least three areas. First, an accurate mathematical model of the Loran-C control system needs to be developed. This will provide the opportunity to easily evaluate possible improvements to Loran-C while minimizing the operational testing required under normal situations. Second, the PID controller needs to be fully operationally tested to determine what further tuning is required prior to implementation. This will provide a suitable bridge between what exists today and the controller of the future. The PID controller will remove the necessity for the Loran-C station operators to second-guess the system. Finally, the Kalman filter needs to be developed further in order to become fully operational. What is presented in this thesis is a proof of concept and no attempt was made at optimizing this algorithm. To enable optimization of the Kalman filter, a wider variety of data sets from all operational Loran-C chains is required. The culmination of this work will be to have a Kalman filter implemented that can respond to the Loran-C changing operational environment while providing optimal navigational information.

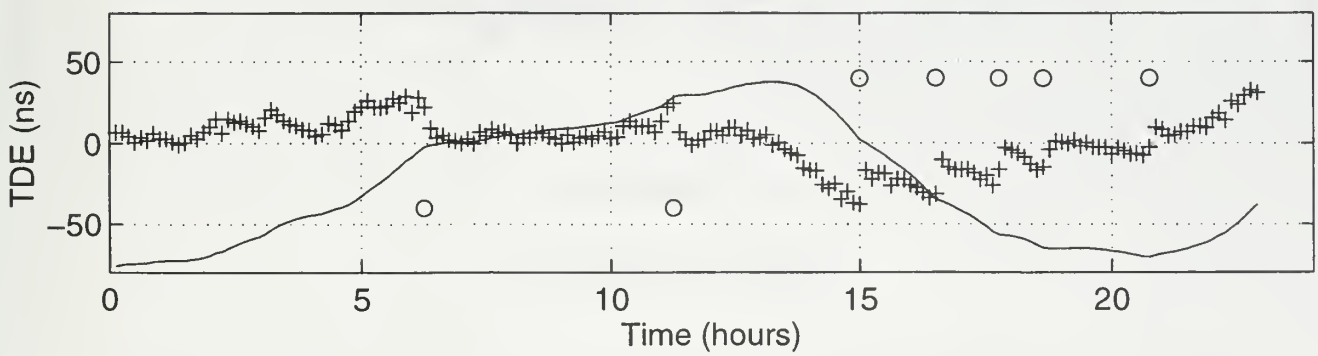
APPENDIX A. STRIP CHARTS GENERATED BY CALOC DATA

1. SOUTH EAST UNITED STATES (SEUS)

a. 18 January 1997



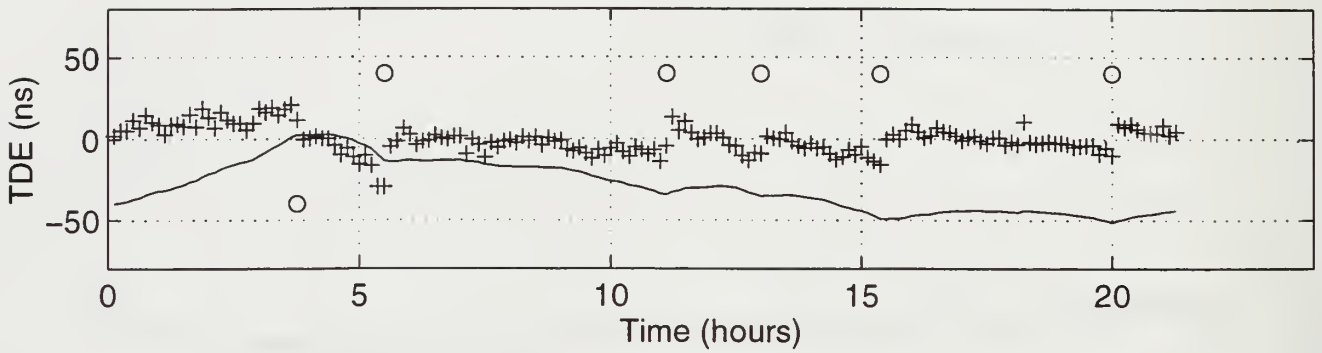
(a) SEUS Station Yankee



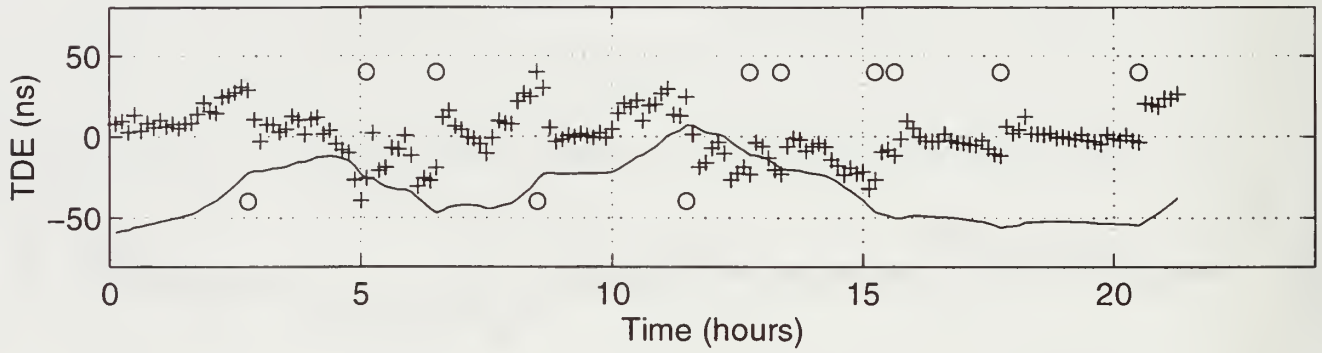
(b) SEUS Station Zulu

Figure 27. CALOC Strip Chart for South East United States (SEUS) Loran-C Chain recorded on 18 January 1997: o Linear Phase Adjustment (LPA) (magnitude times 2); + Time Difference Error (TDE); — Cumulative TDE.

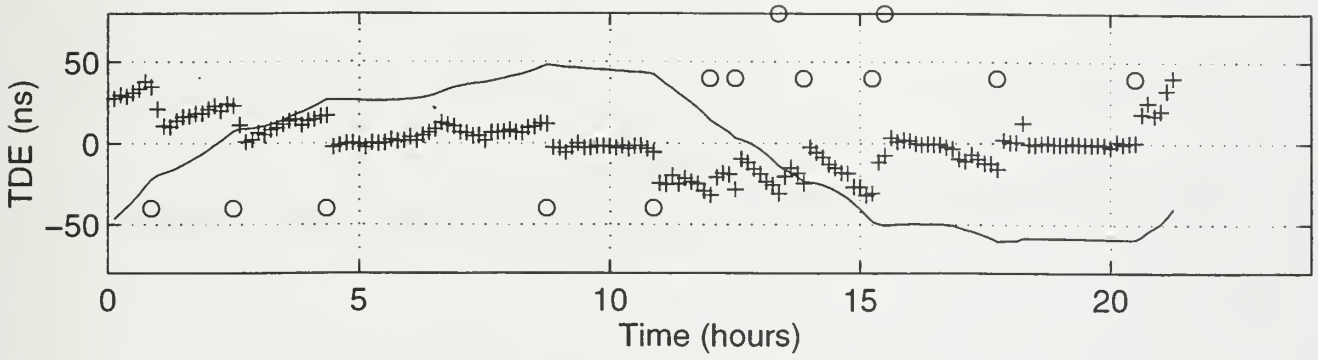
b. 20 January 1997



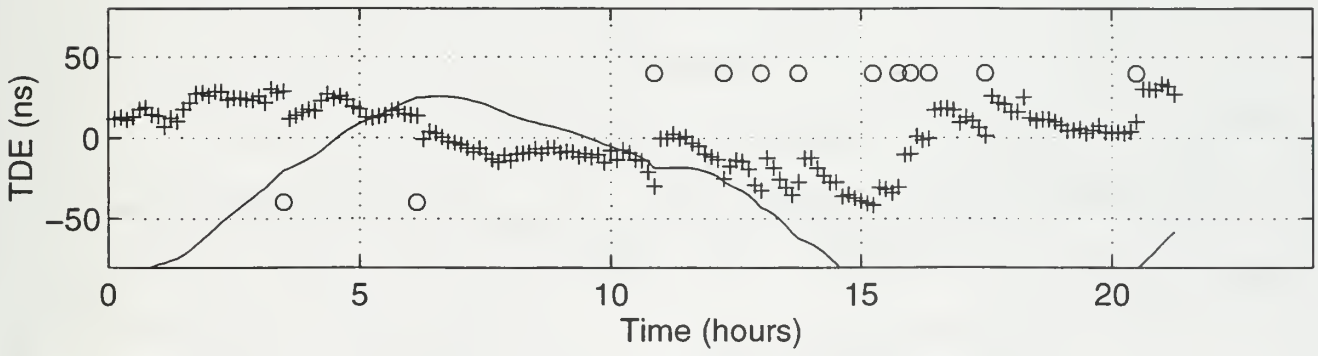
(a) SEUS Station Whiskey



(b) SEUS Station Xray



(c) SEUS Station Yankee

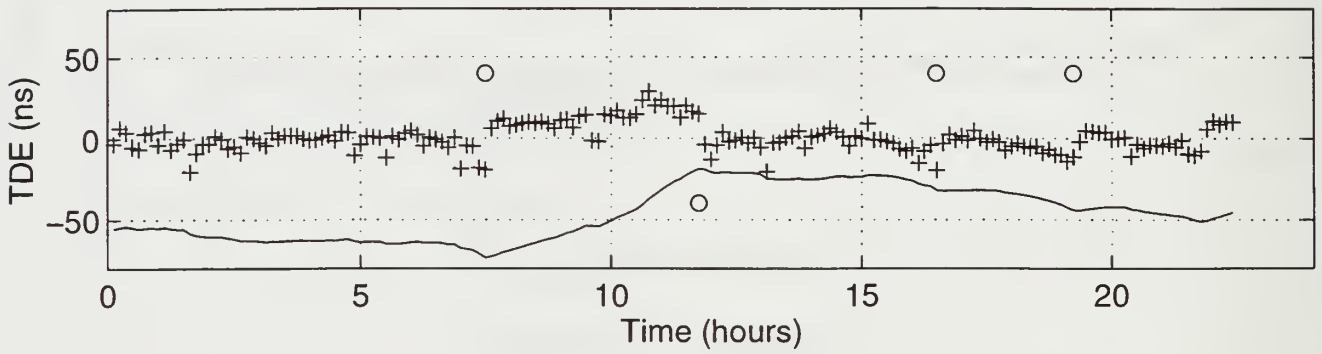


(d) SEUS Station Zulu

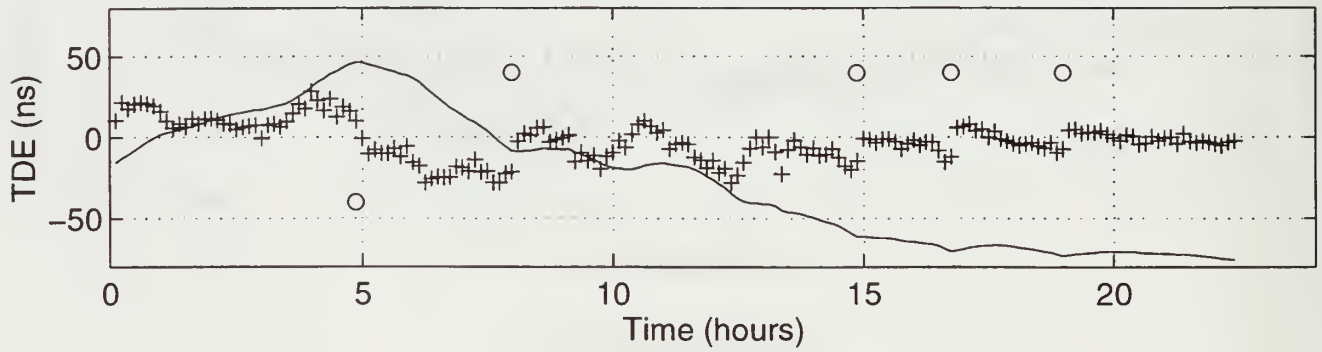
Figure 28. CALOC Strip Chart for South East United States (SEUS) Loran-C Chain recorded on 20 January 1997: o Linear Phase Adjustment (LPA) (magnitude times 2); + Time Difference Error (TDE); — Cumulative TDE.

2. SOUTH CENTRAL UNITED STATES (SOCUS)

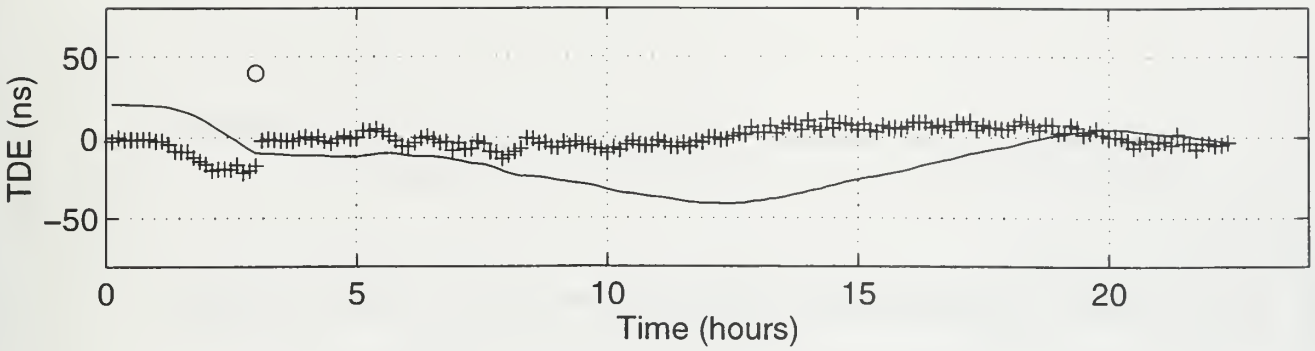
a. 18 January 1997



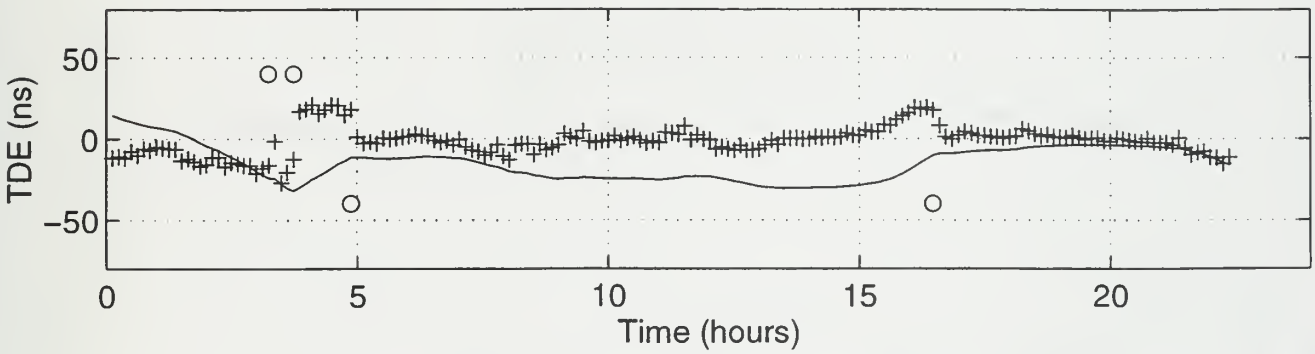
(a) SOCUS Station Victor



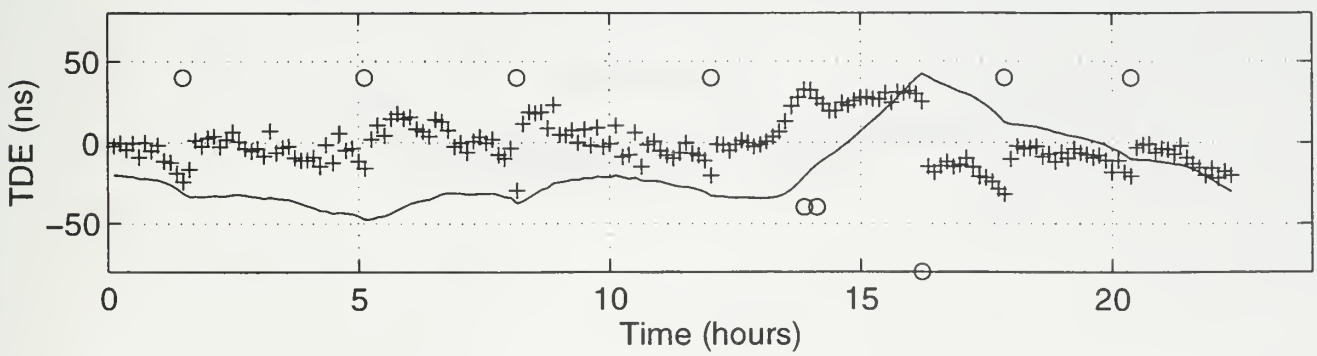
(b) SOCUS Station Whiskey



(c) SOCUS Station Xray



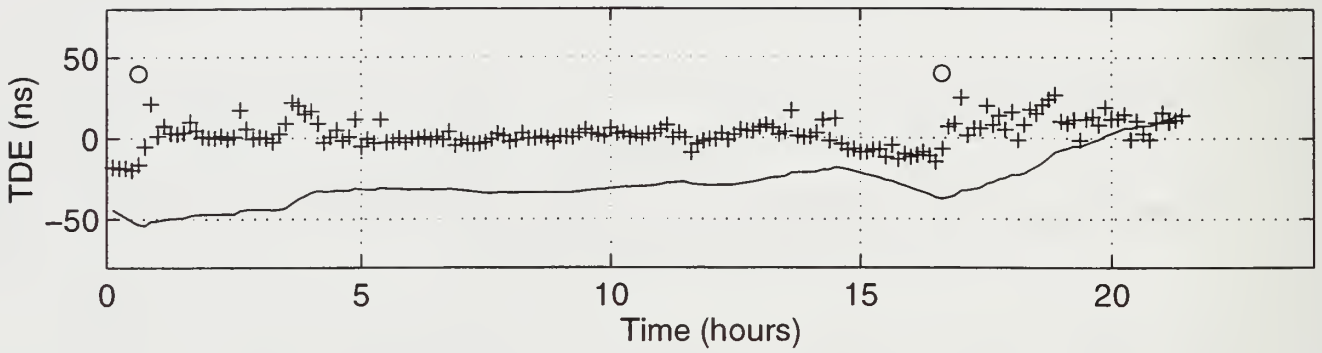
(d) SOCUS Station Yankee



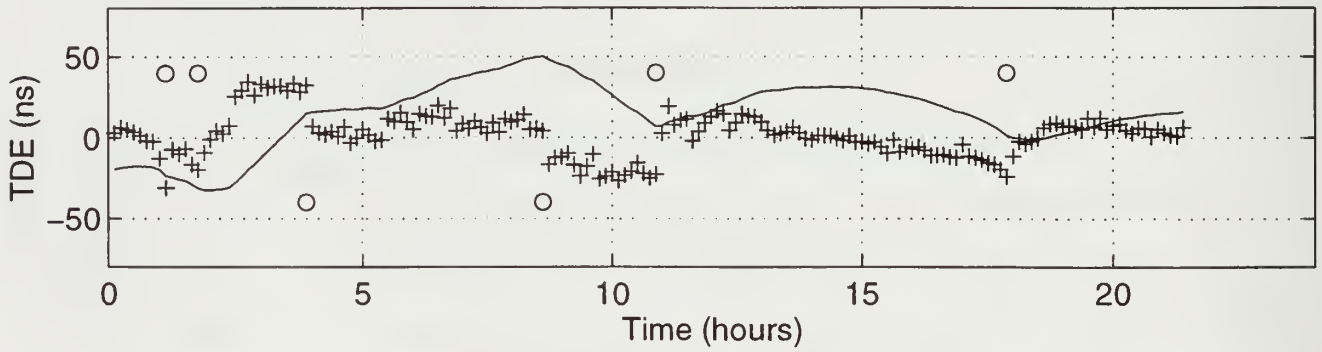
(e) SOCUS Station Zulu

Figure 29. CALOC Strip Chart for South Central United States (SOCUS) Loran-C Chain recorded on 18 January 1997: o Linear Phase Adjustment (LPA) (magnitude times 2); + Time Difference Error (TDE); — Cumulative TDE.

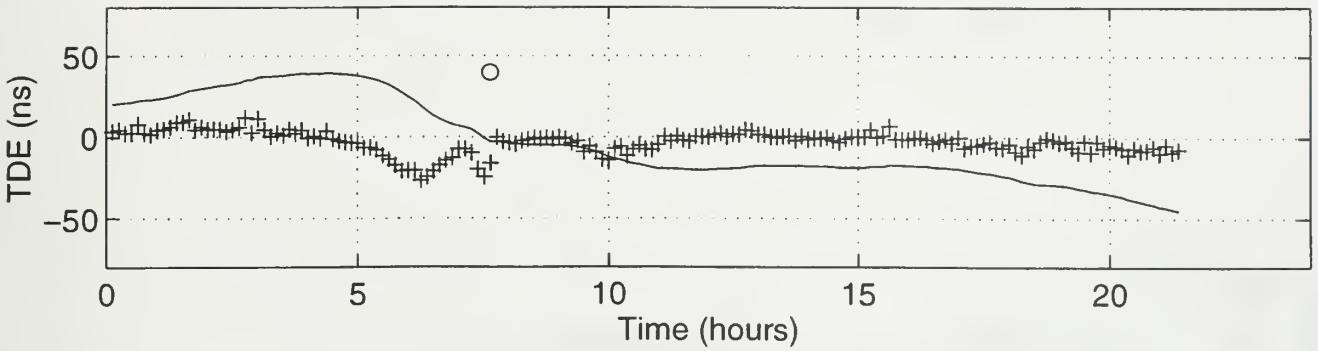
b. 20 January 1997



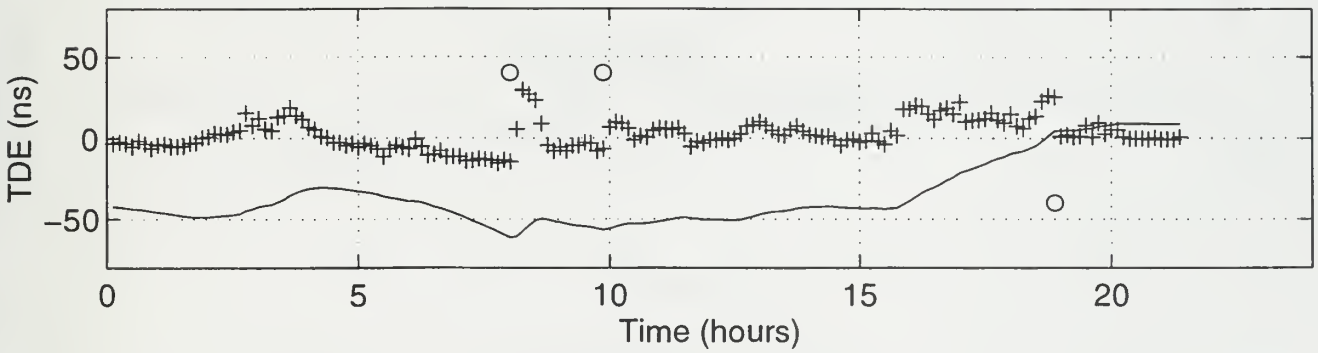
(a) SOCUS Station Victor



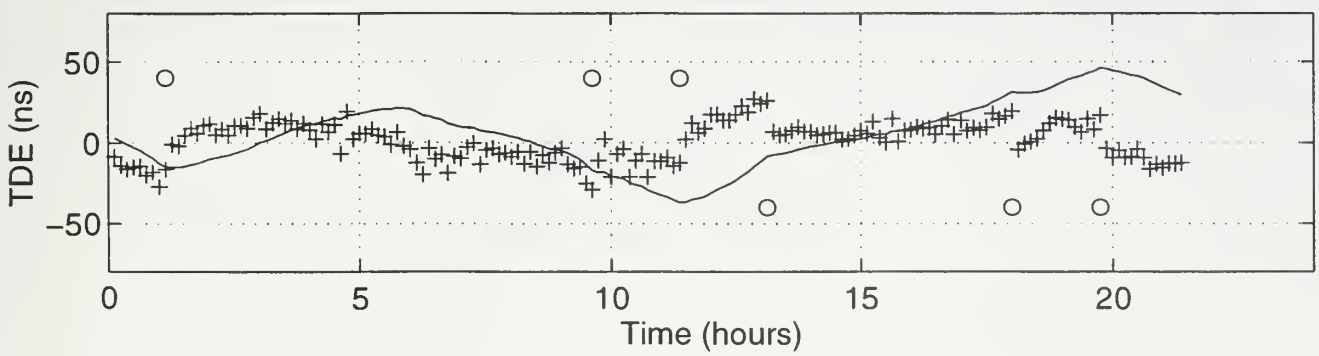
(b) SOCUS Station Whiskey



(c) SOCUS Station Xray



(d) SOCUS Station Yankee

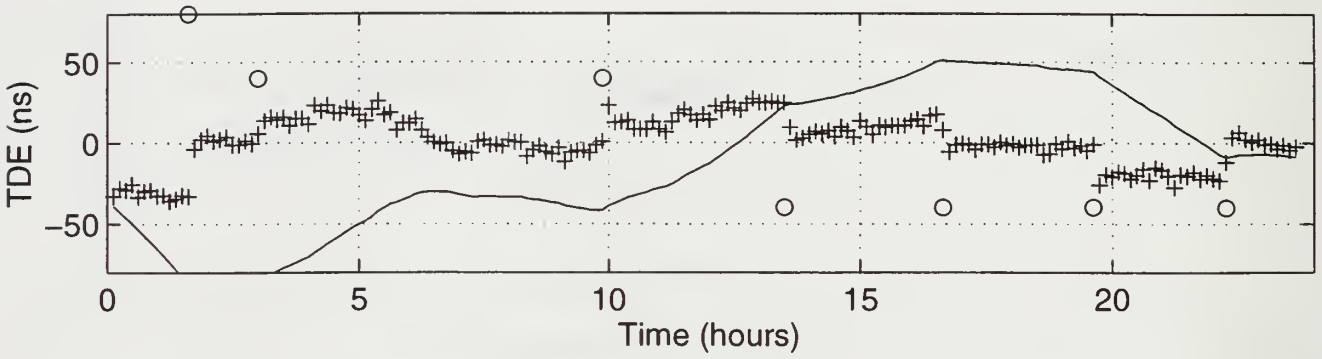


(e) SOCUS Station Zulu

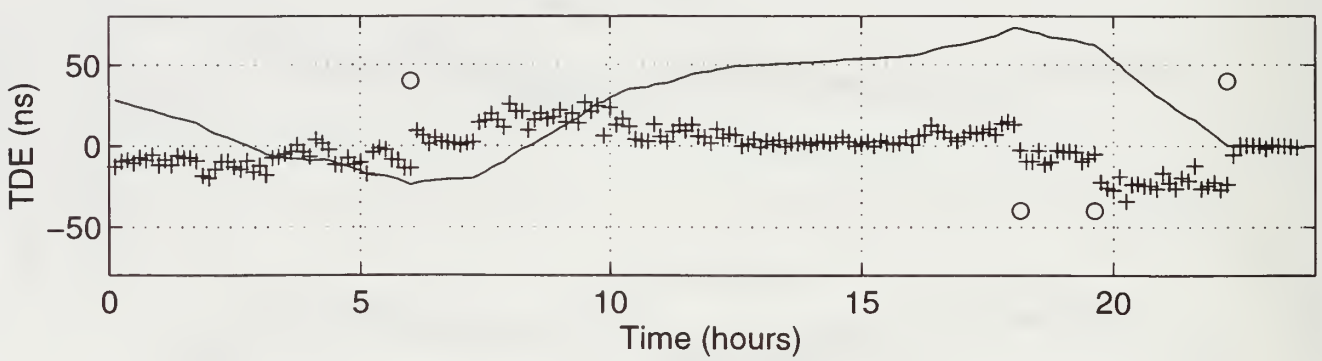
Figure 30. CALOC Strip Chart for South Central United States (SOCUS) Loran-C Chain recorded on 20 January 1997: o Linear Phase Adjustment (LPA) (magnitude times 2); + Time Difference Error (TDE); — Cumulative TDE.

3. NORTH CENTRAL UNITED STATES (NOCUS)

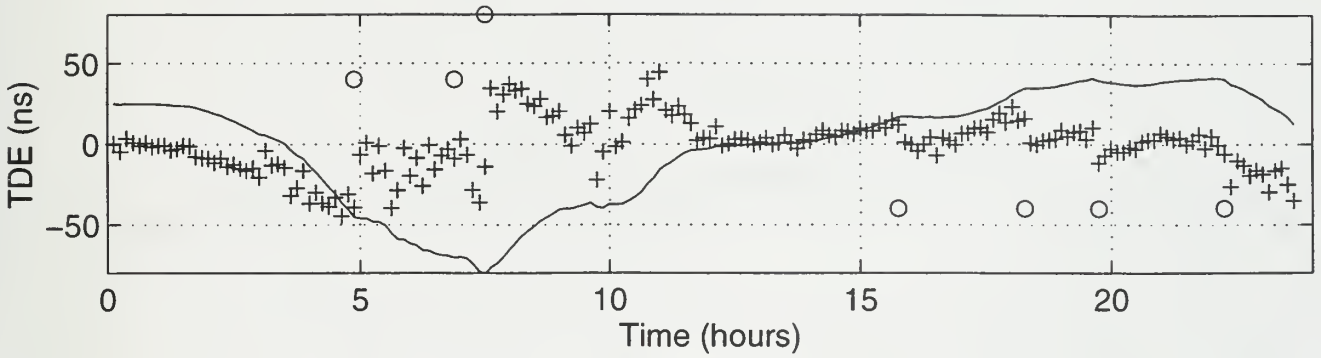
a. 4 May 1997



(a) NOCUS Station Whiskey



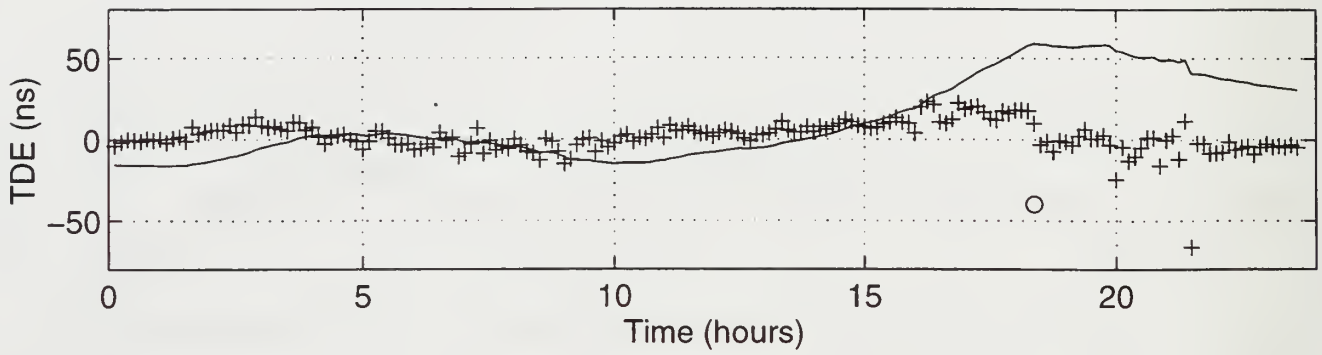
(b) NOCUS Station Xray



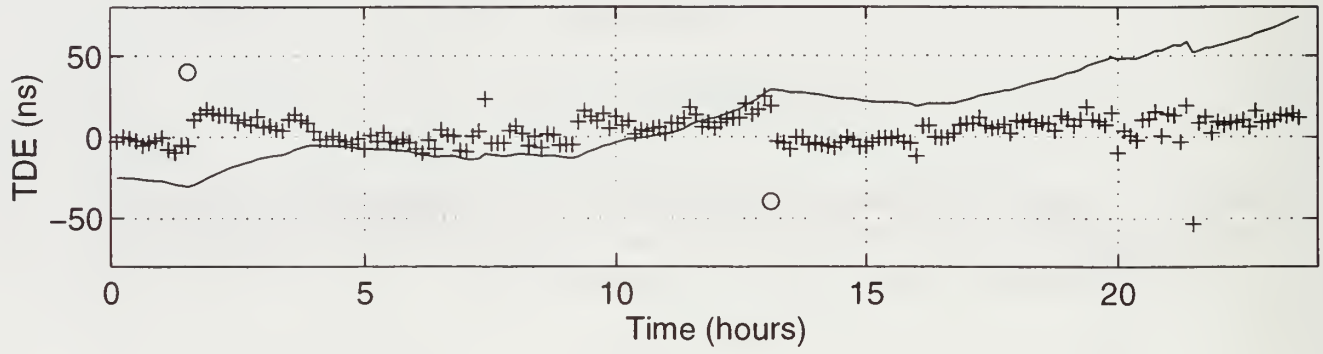
(c) NOCUS Station Yankee

Figure 31. CALOC Strip Chart for North Central United States (NOCUS) Loran-C Chain recorded on 4 May 1997: o Linear Phase Adjustment (LPA) (magnitude times 2); + Time Difference Error (TDE); — Cumulative TDE.

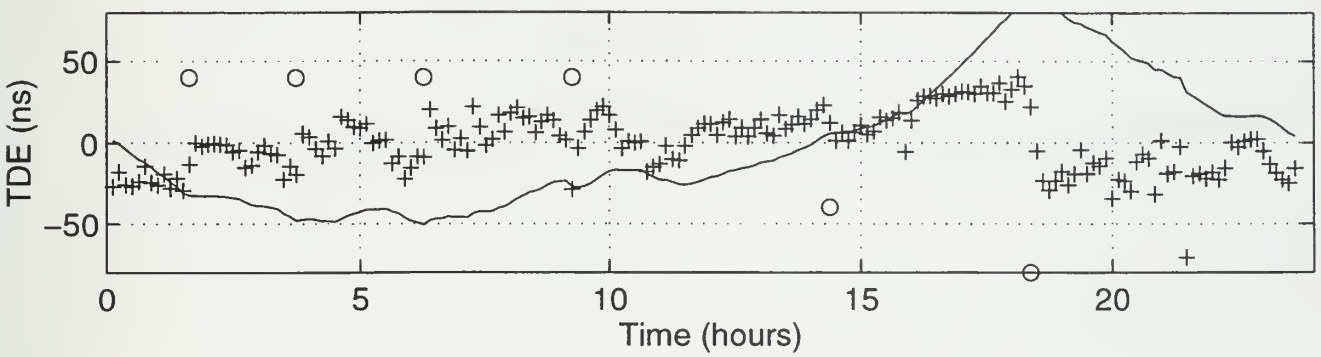
b. 5 May 1997



(a) NOCUS Station Whiskey



(b) NOCUS Station Xray

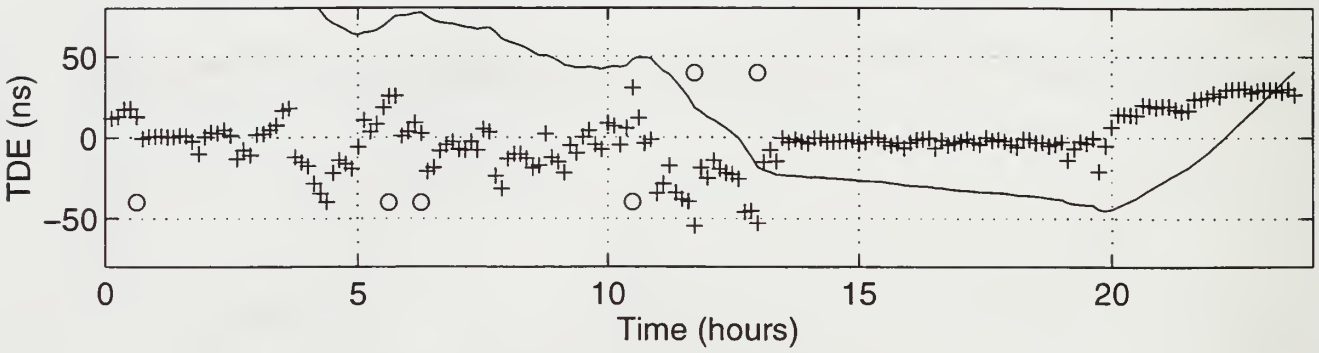


(c) NOCUS Station Yankee

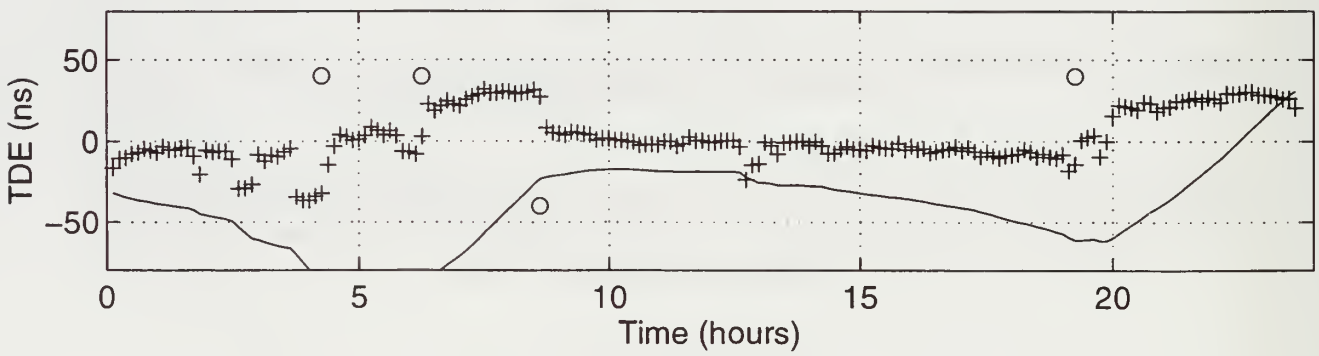
Figure 32. CALOC Strip Chart for North Central United States (NOCUS) Loran-C Chain recorded on 5 May 1997: o Linear Phase Adjustment (LPA) (magnitude times 2); + Time Difference Error (TDE); — Cumulative TDE.

4. UNITED STATES WEST COAST (USWC)

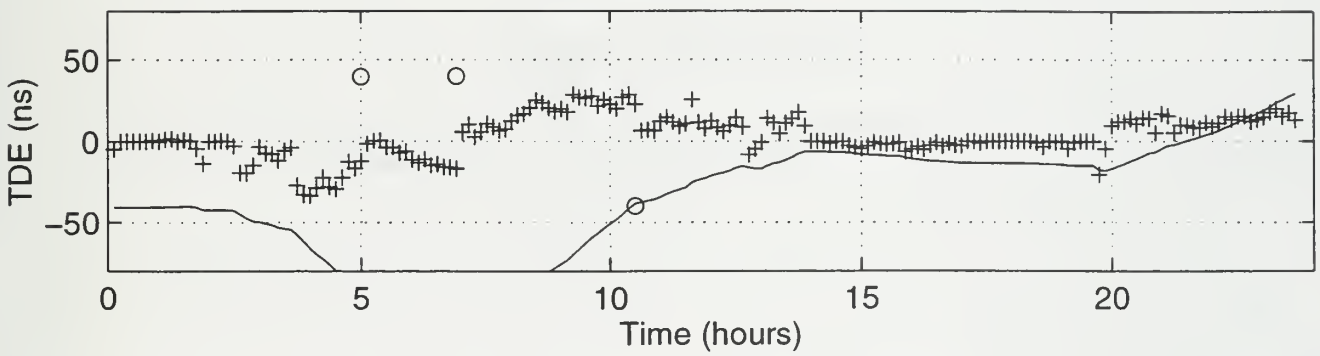
a. 4 May 1997



(a) USWC Station Whiskey



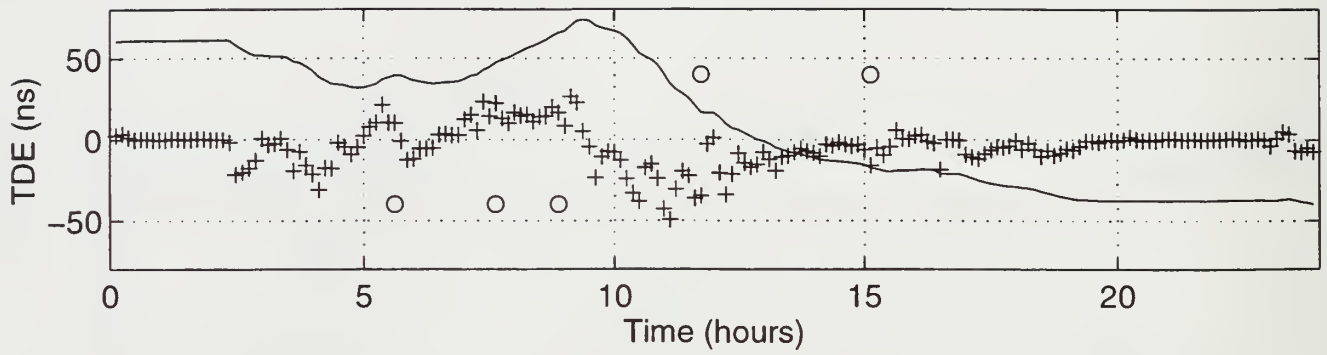
(b) USWC Station Xray



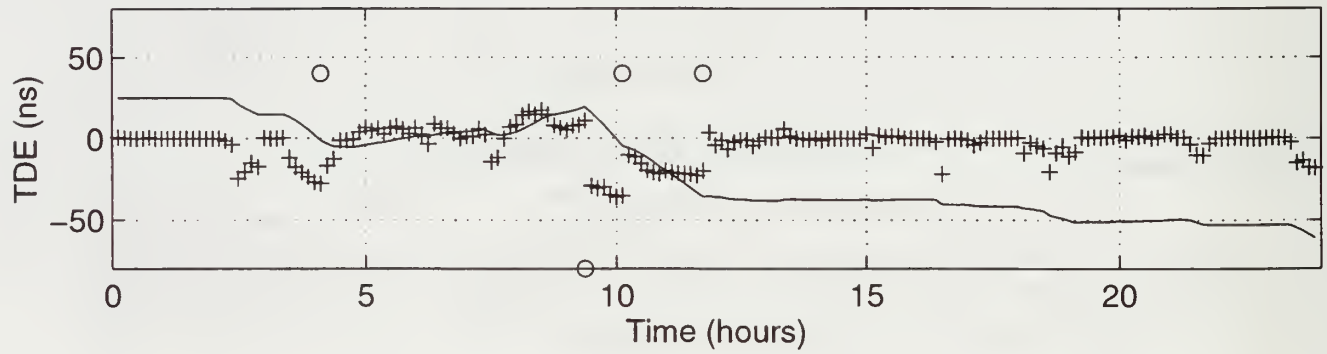
(c) USWC Station Yankee

Figure 33. CALOC Strip Chart for United States West Coast (USWC) Loran-C Chain recorded on 4 May 1997: o Linear Phase Adjustment (LPA) (magnitude times 2); + Time Difference Error (TDE); — Cumulative TDE.

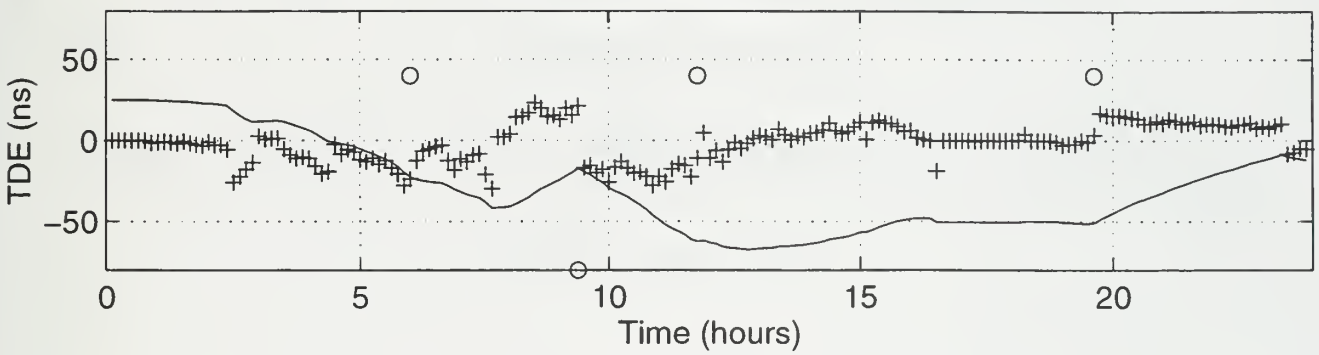
b. 5 May 1997



(a) USWC Station Whiskey



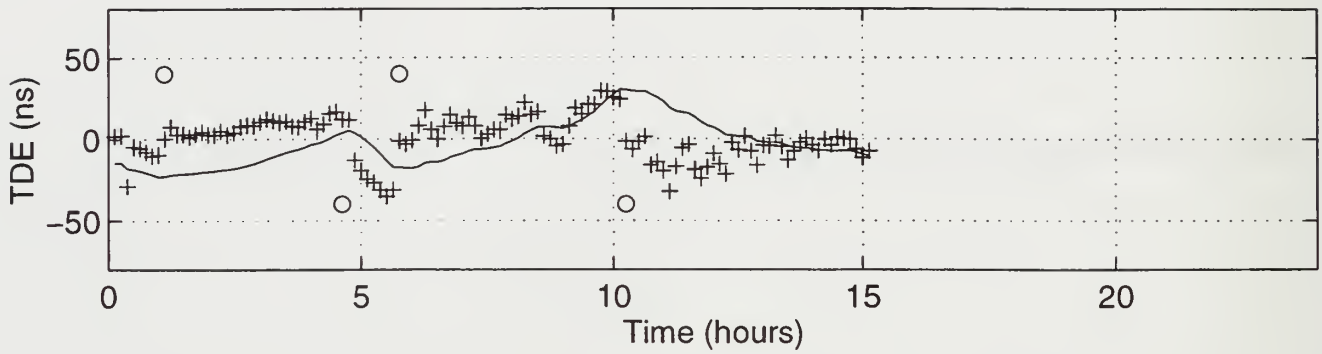
(b) USWC Station Xray



(c) USWC Station Yankee

Figure 34. CALOC Strip Chart for United States West Coast (USWC) Loran-C Chain recorded on 5 May 1997: o Linear Phase Adjustment (LPA) (magnitude times 2); + Time Difference Error (TDE); — Cumulative TDE.

5. KODIAK



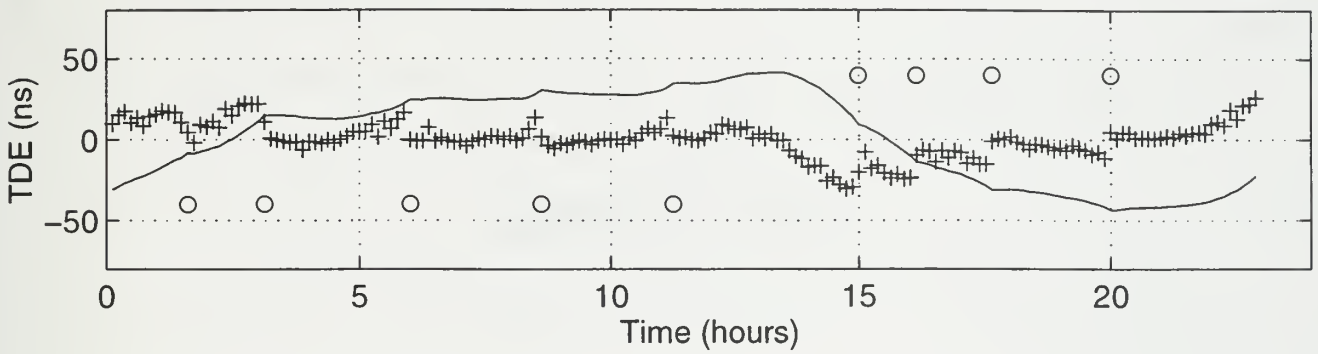
(a) Kodiak Station Zulu

Figure 35. CALOC Strip Chart for Kodiak Loran-C Chain: o Linear Phase Adjustment (LPA) (magnitude times 2); + Time Difference Error (TDE); — Cumulative TDE.

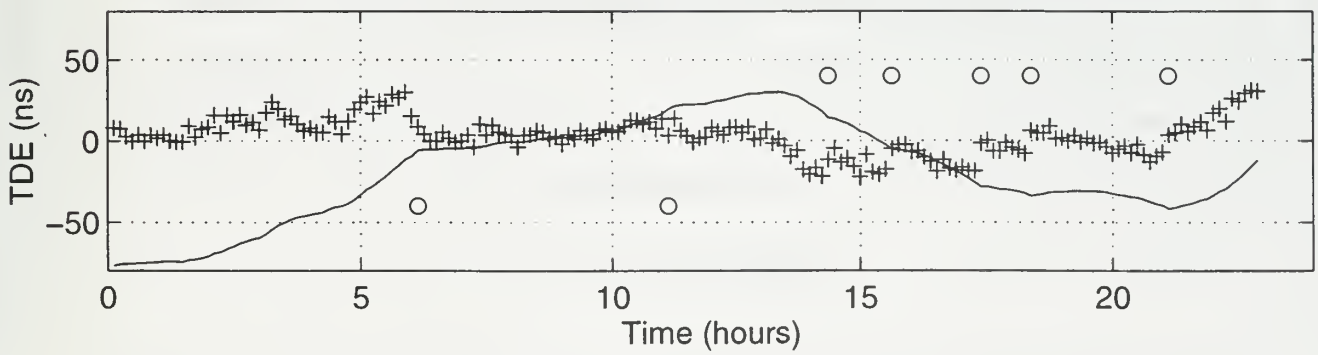
APPENDIX B. STRIP CHARTS GENERATED BY PID CONTROLLER

1. SOUTH EAST UNITED STATES (SEUS)

a. 18 January 1997



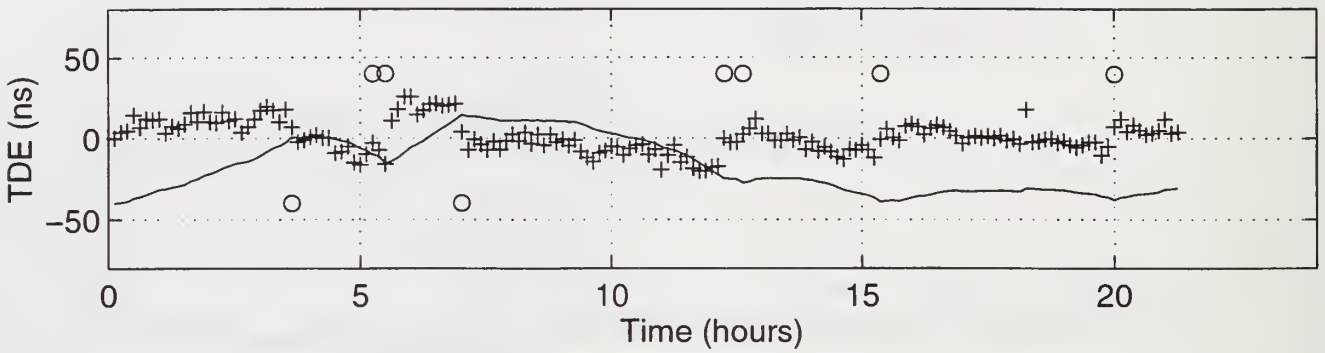
(a) SEUS Station Yankee



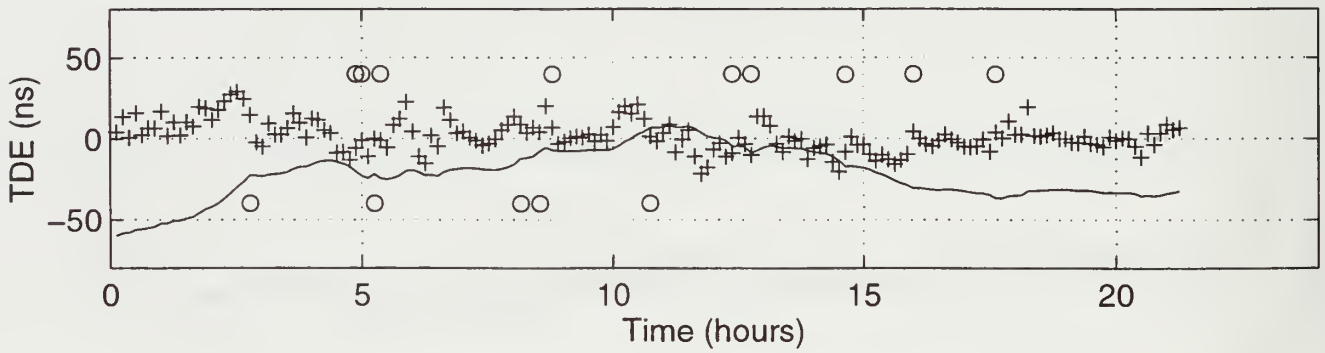
(b) SEUS Station Zulu

Figure 36. PID Controller Strip Chart for South East United States (SEUS) Loran-C Chain recorded on 18 January 1997: o Linear Phase Adjustment (LPA) (magnitude times 2); + Time Difference Error (TDE); — Cumulative TDE.

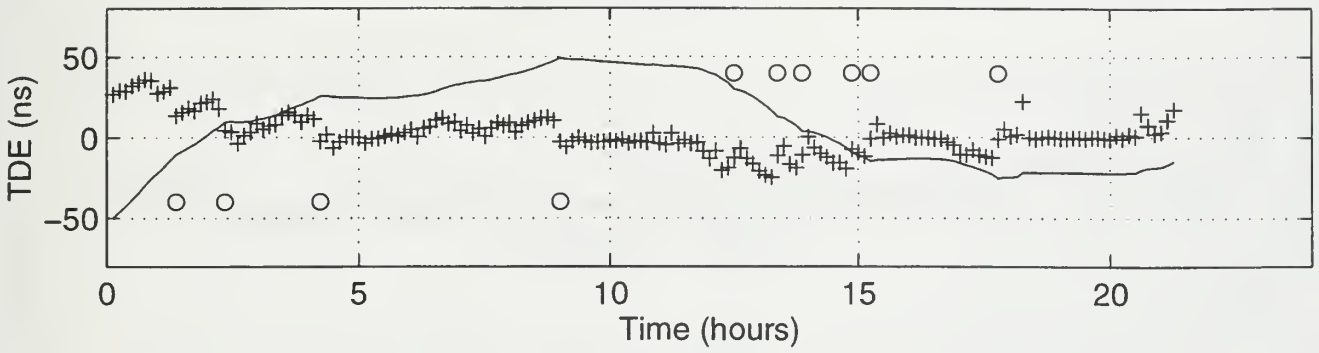
b. 20 January 1997



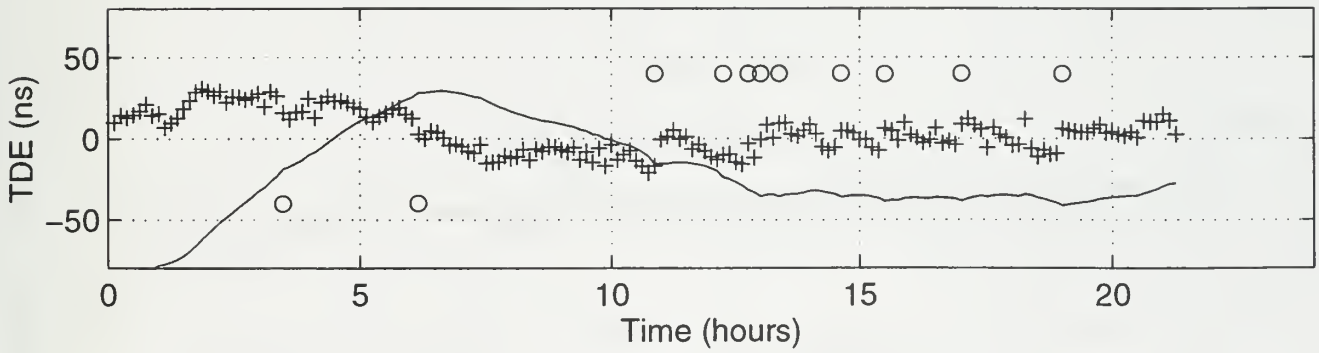
(a) SEUS Station Whiskey



(b) SEUS Station Xray



(c) SEUS Station Yankee

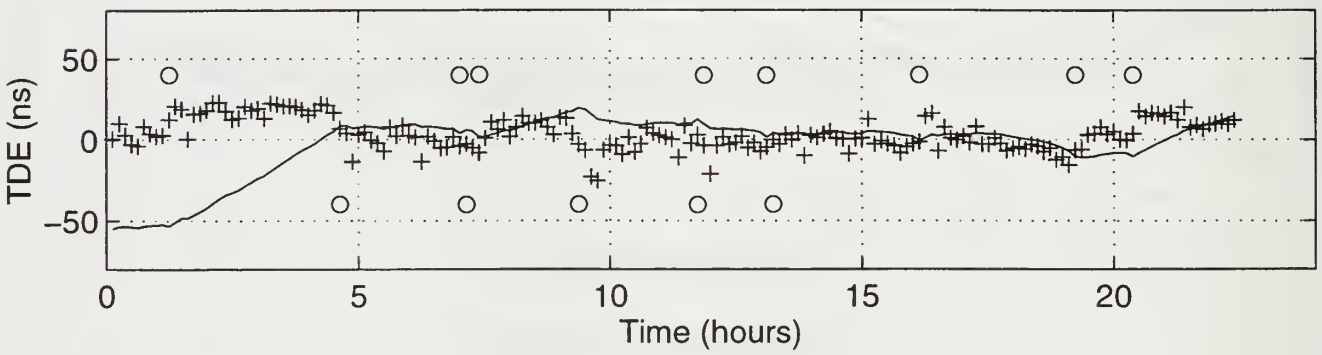


(d) SEUS Station Zulu

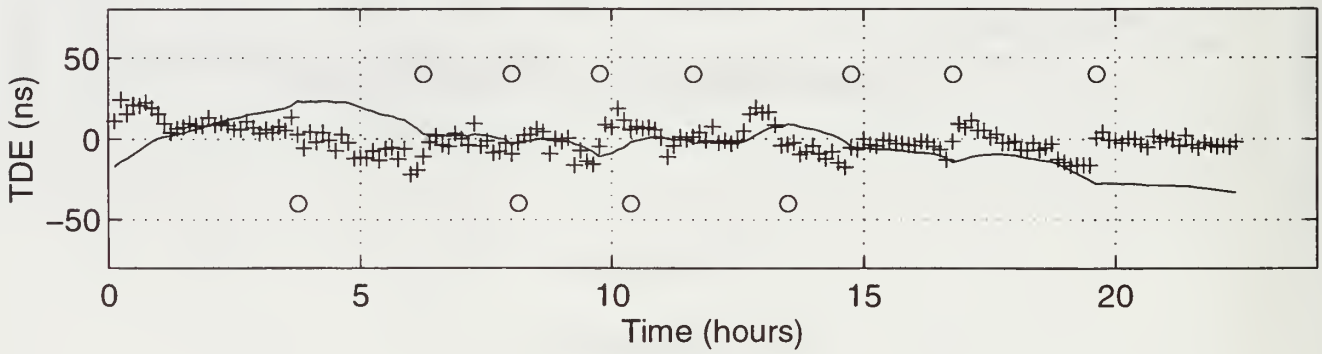
Figure 37. PID Controller Strip Chart for South East United States (SEUS) Loran-C Chain recorded on 20 January 1997: o Linear Phase Adjustment (LPA) (magnitude times 2); + Time Difference Error (TDE); — Cumulative TDE.

2. SOUTH CENTRAL UNITED STATES (SOCUS)

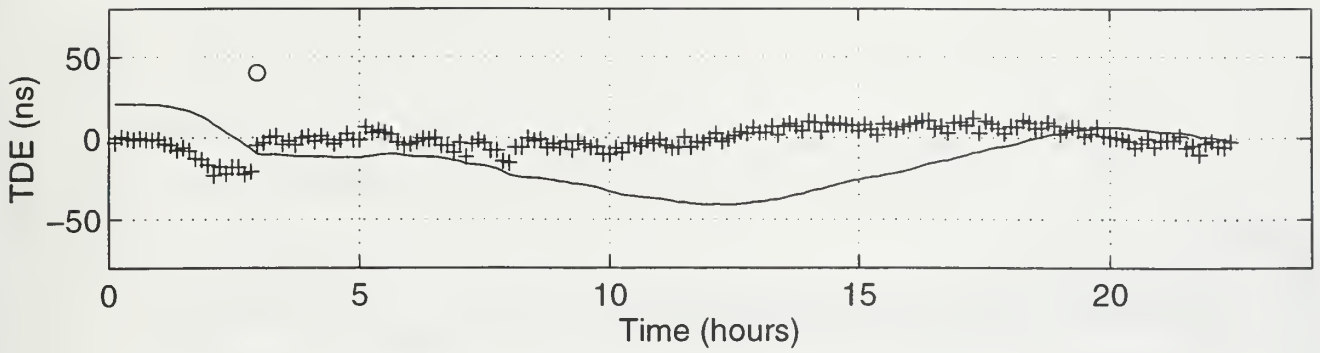
a. 18 January 1997



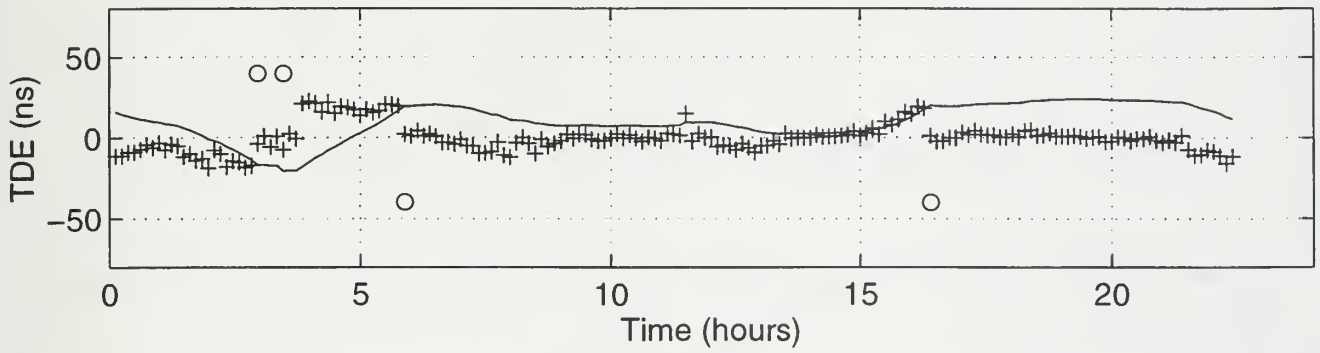
(a) SOCUS Station Victor



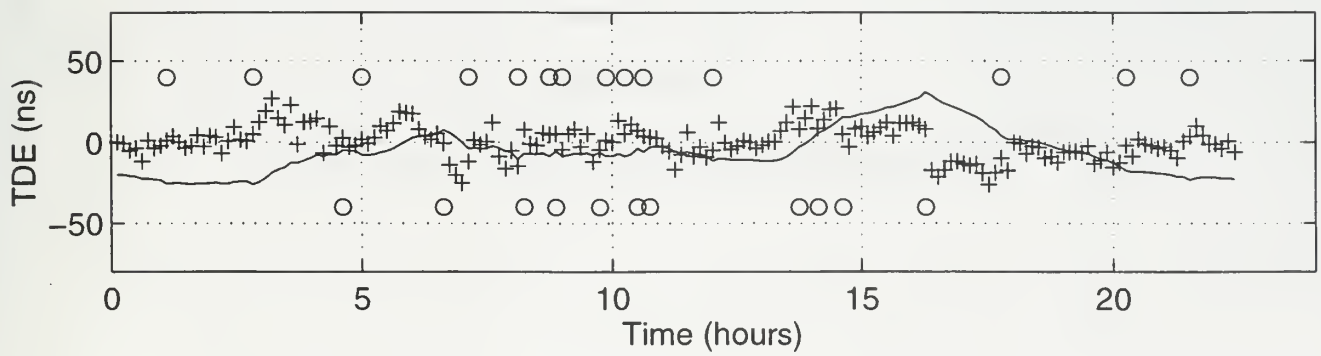
(b) SOCUS Station Whiskey



(c) SOCUS Station Xray



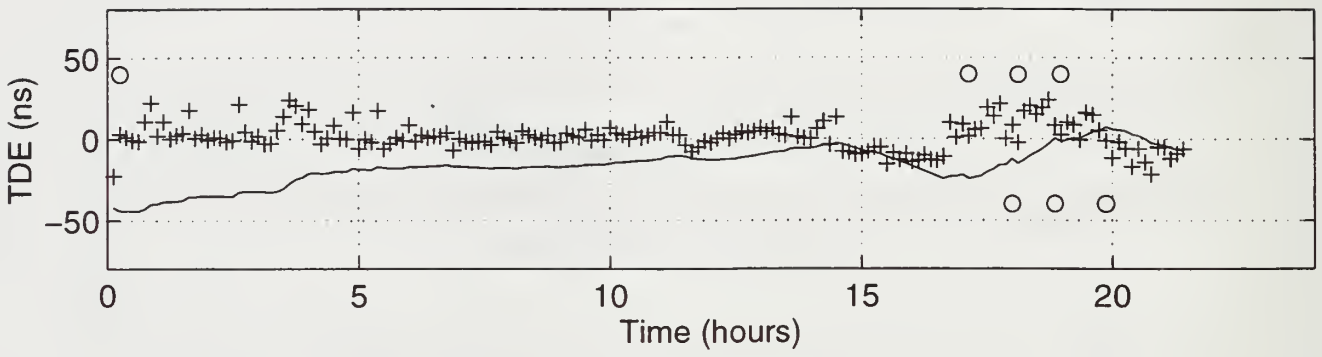
(d) SOCUS Station Yankee



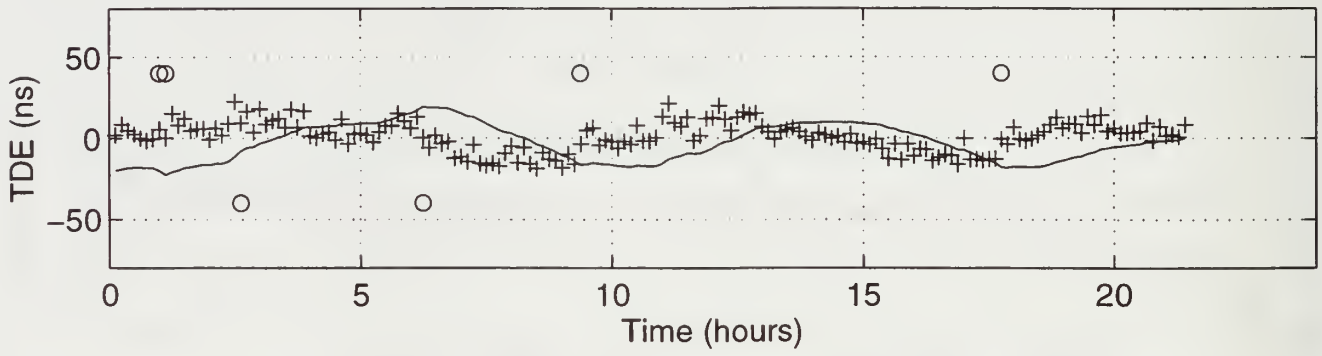
(e) SOCUS Station Zulu

Figure 38. PID Controller Strip Chart for South Central United States (SOCUS) Loran-C Chain recorded on 18 January 1997: o Linear Phase Adjustment (LPA) (magnitude times 2); + Time Difference Error (TDE); — Cumulative TDE.

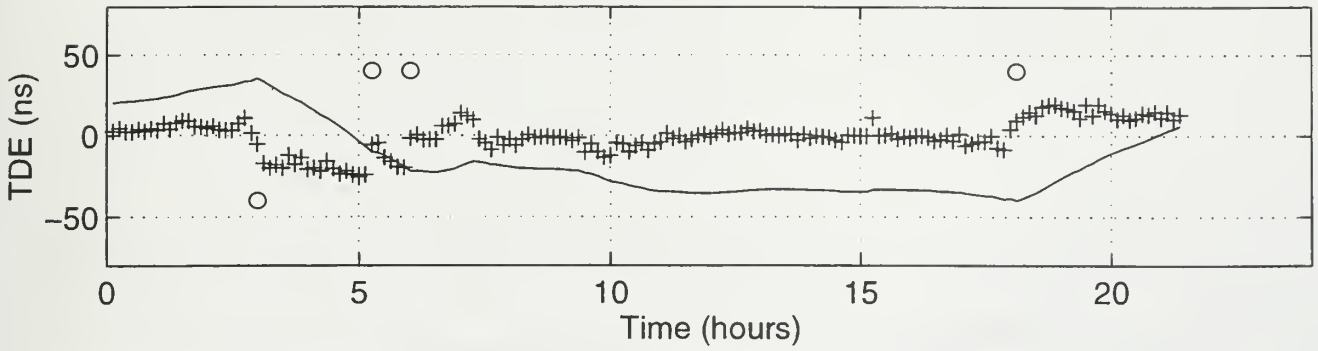
b. 20 January 1997



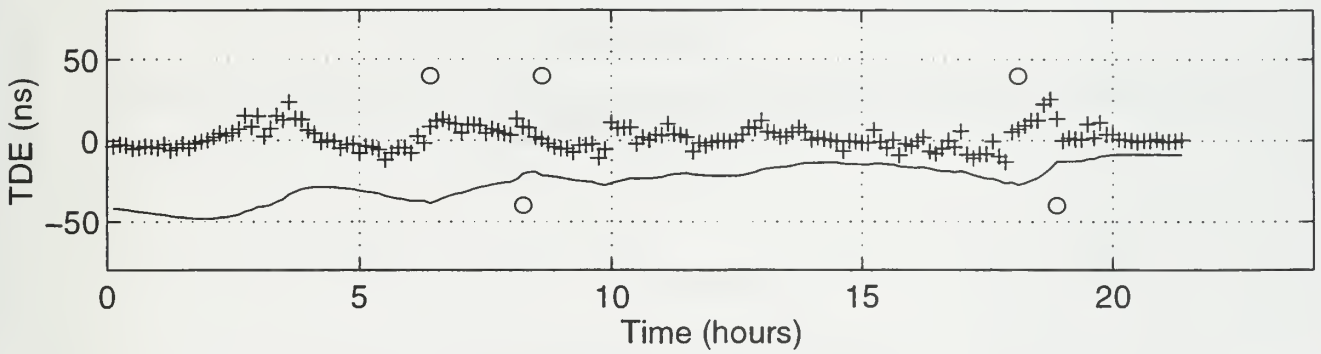
(a) SOCUS Station Victor



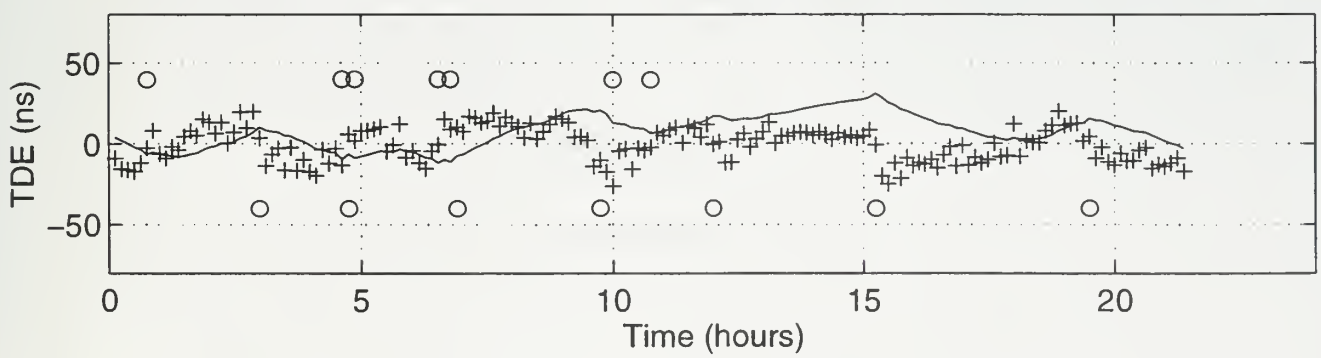
(b) SOCUS Station Whiskey



(c) SOCUS Station Xray



(d) SOCUS Station Yankee

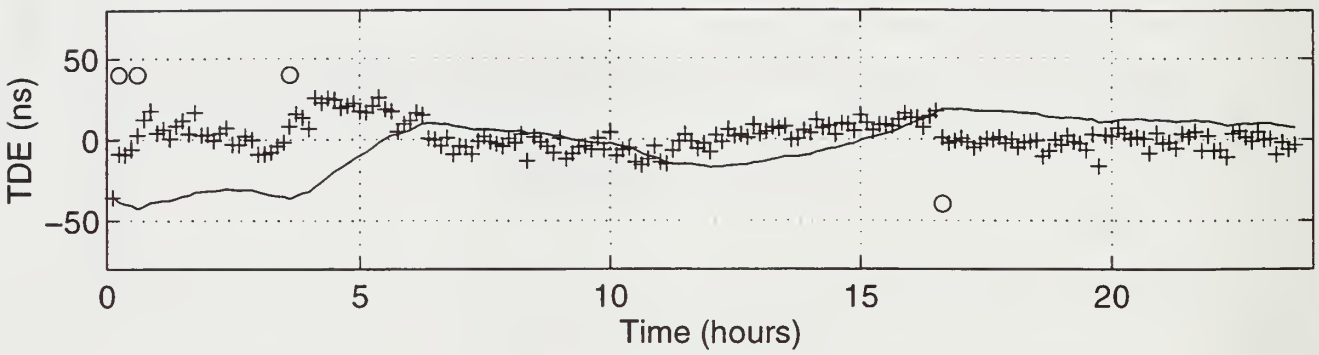


(e) SOCUS Station Zulu

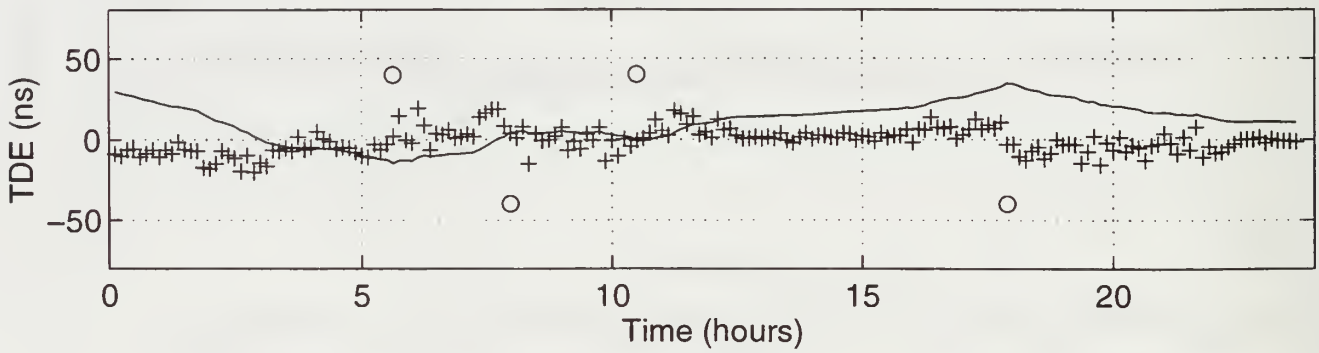
Figure 39. PID Controller Strip Chart for South Central United States (SOCUS) Loran-C Chain recorded on 20 January 1997: o Linear Phase Adjustment (LPA) (magnitude times 2); + Time Difference Error (TDE); — Cumulative TDE.

3. NORTH CENTRAL UNITED STATES (NOCUS)

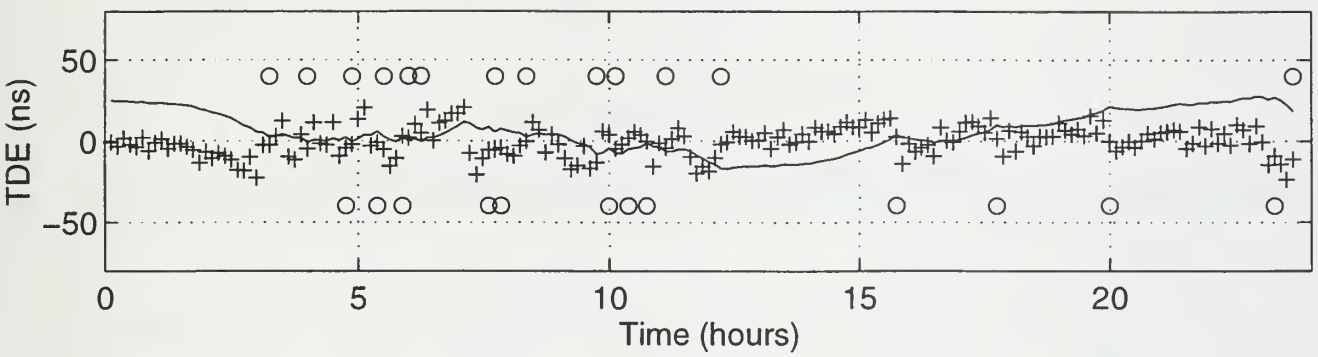
a. 4 May 1997



(a) NOCUS Station Whiskey



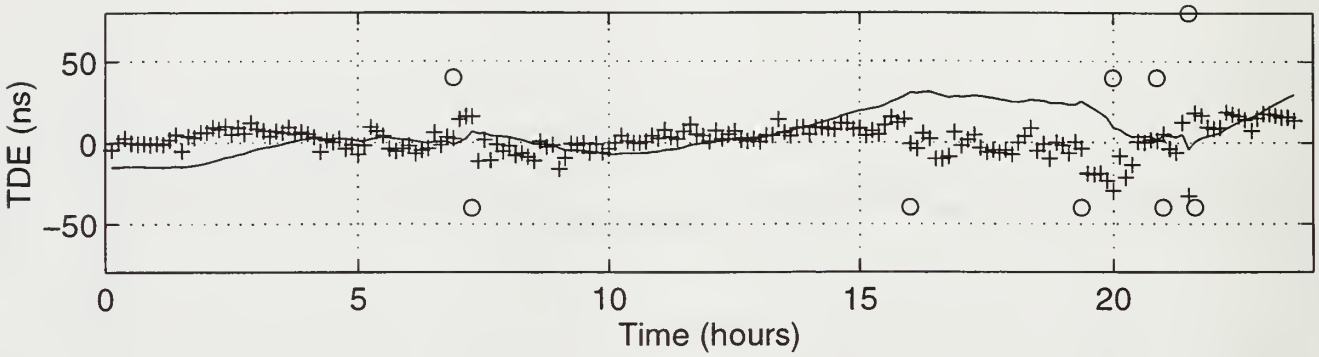
(b) NOCUS Station Xray



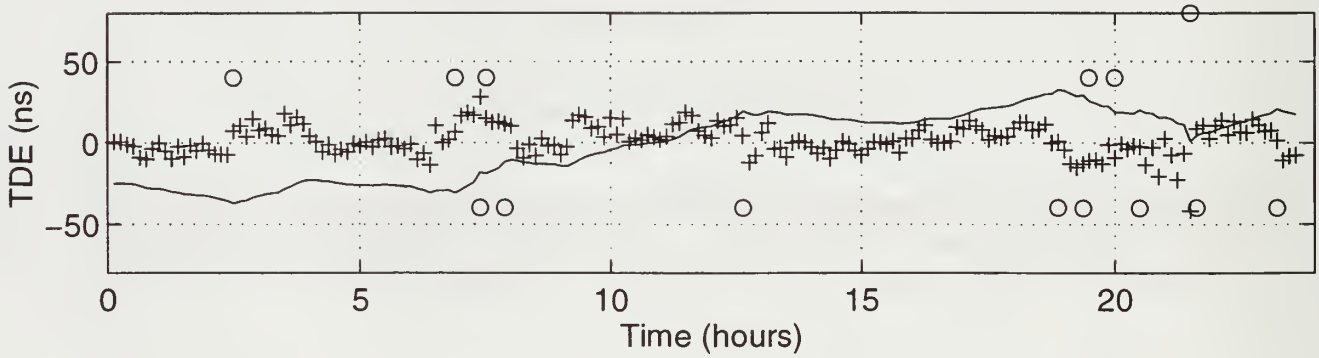
(c) NOCUS Station Yankee

Figure 40. PID Controller Strip Chart for North Central United States (NOCUS) Loran-C Chain recorded on 4 May 1997: o Linear Phase Adjustment (LPA) (magnitude times 2); + Time Difference Error (TDE); — Cumulative TDE.

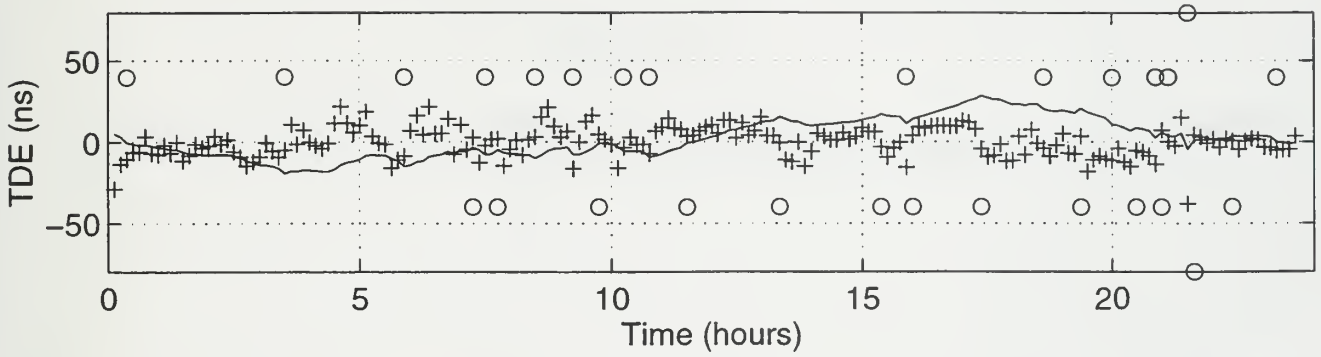
b. 5 May 1997



(a) NOCUS Station Whiskey



(b) NOCUS Station Xray

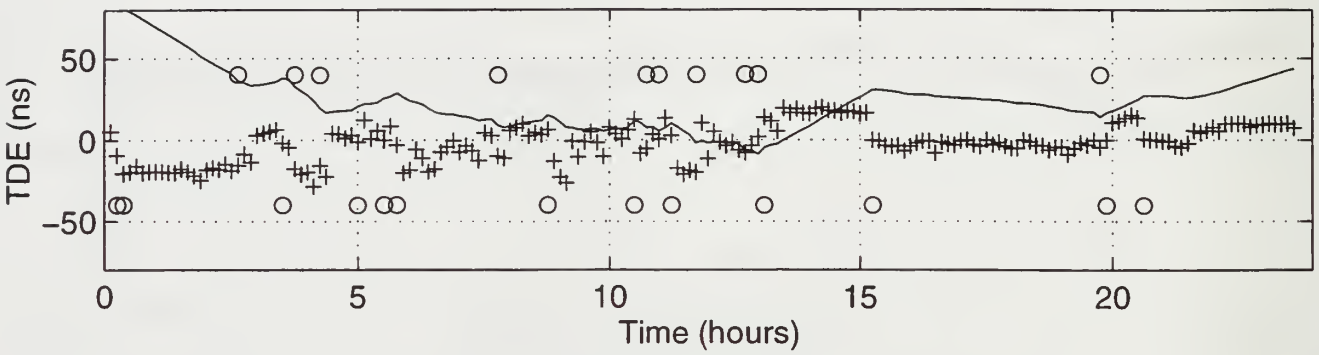


(c) NOCUS Station Yankee

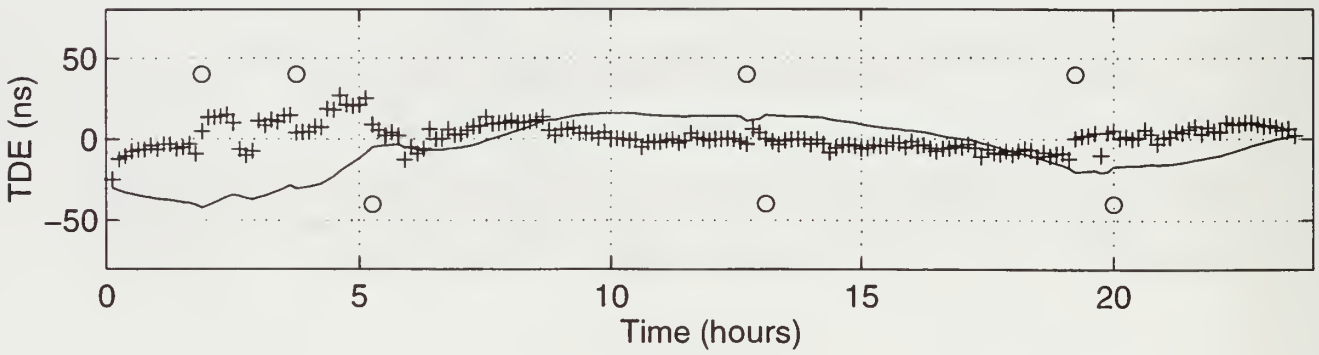
Figure 41. PID Controller Strip Chart for North Central United States (NOCUS) Loran-C Chain recorded on 5 May 1997: o Linear Phase Adjustment (LPA) (magnitude times 2); + Time Difference Error (TDE); — Cumulative TDE.

4. UNITED STATES WEST COAST (USWC)

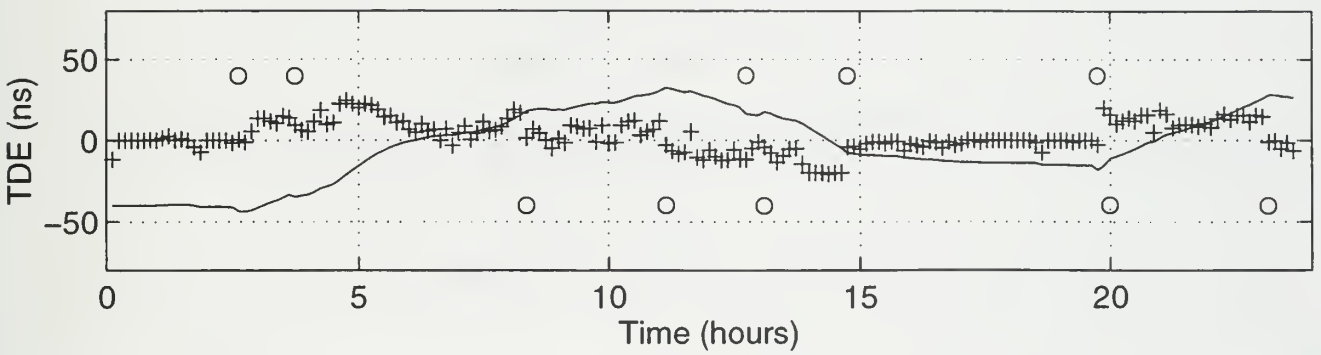
a. 4 May 1997



(a) USWC Station Whiskey



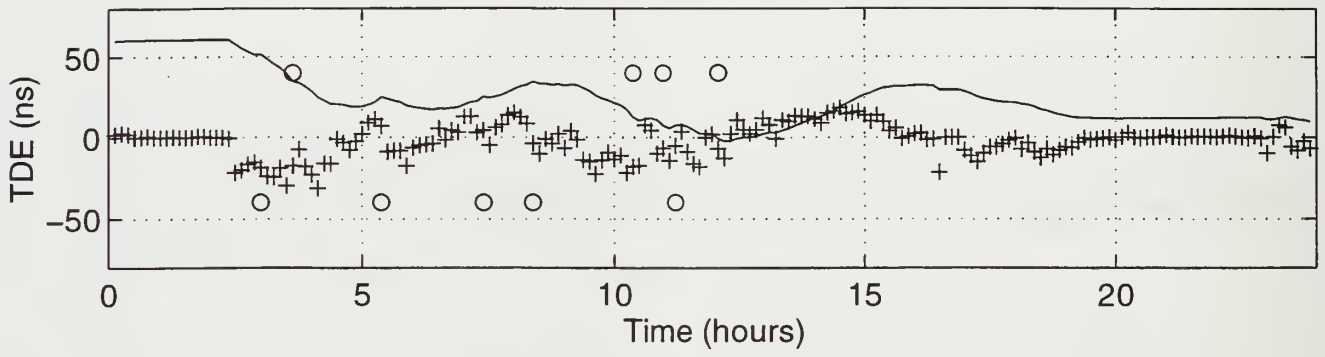
(b) USWC Station Xray



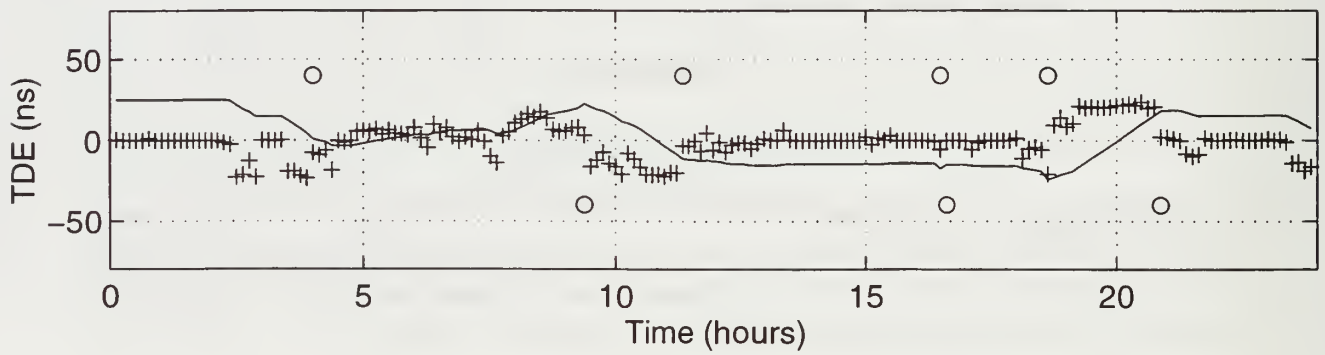
(c) USWC Station Yankee

Figure 42. PID Controller Strip Chart for United States West Coast (USWC) Loran-C Chain recorded on 4 May 1997: o Linear Phase Adjustment (LPA) (magnitude times 2); + Time Difference Error (TDE); — Cumulative TDE.

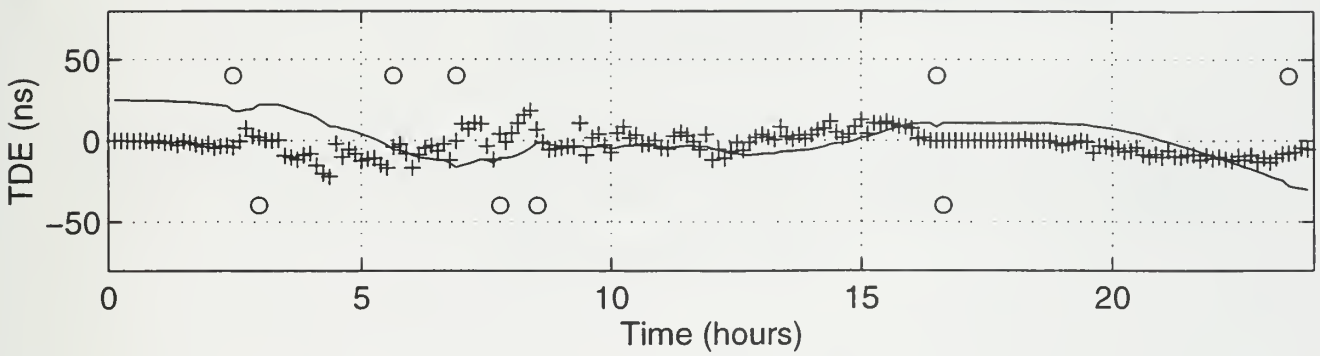
b. 5 May 1997



(a) USWC Station Whiskey



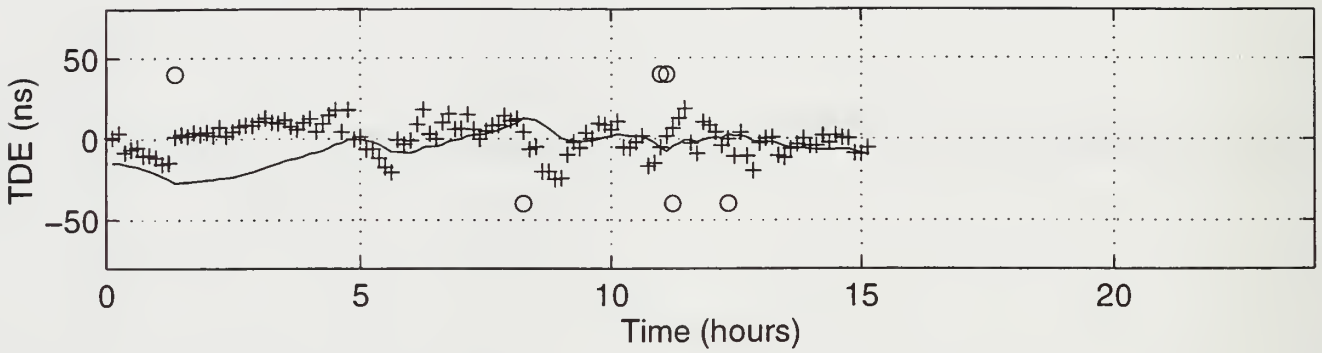
(b) USWC Station Xray



(c) USWC Station Yankee

Figure 43. PID Controller Strip Chart for United States West Coast (USWC) Loran-C Chain recorded on 5 May 1997: o Linear Phase Adjustment (LPA) (magnitude times 2); + Time Difference Error (TDE); — Cumulative TDE.

5. KODIAK



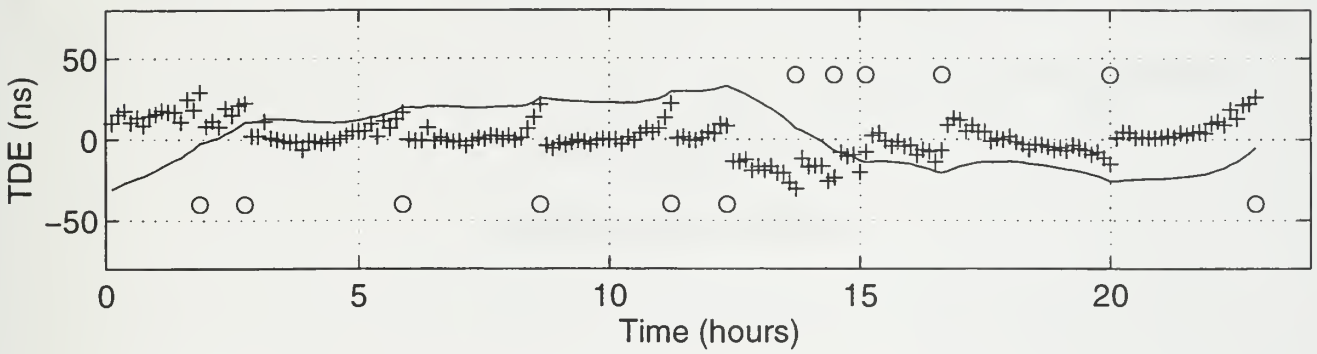
(a) Kodiak Station Zulu

Figure 44. PID Controller Strip Chart for Kodiak Loran-C Chain: **o** Linear Phase Adjustment (LPA) (magnitude times 2); + Time Difference Error (TDE); — Cumulative TDE.

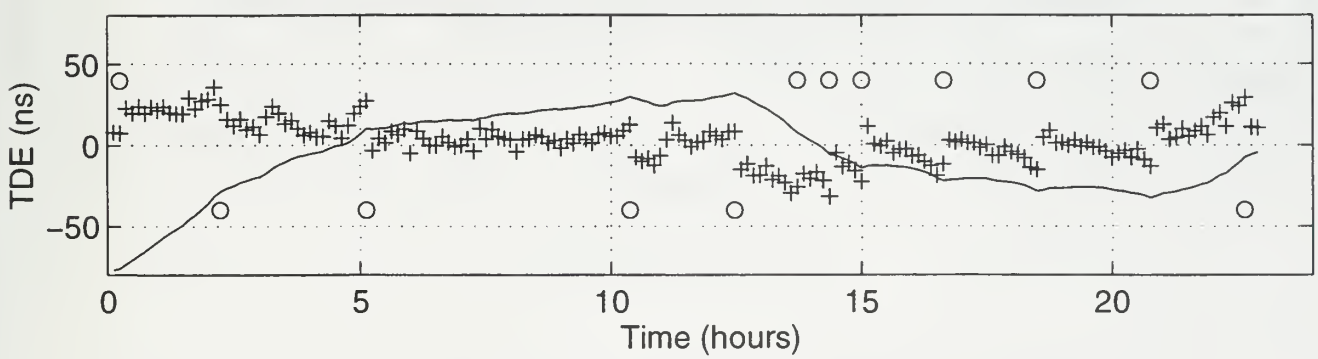
APPENDIX C. STRIP CHARTS GENERATED BY KALMAN FILTER

1. SOUTH EAST UNITED STATES (SEUS)

a. 18 January 1997



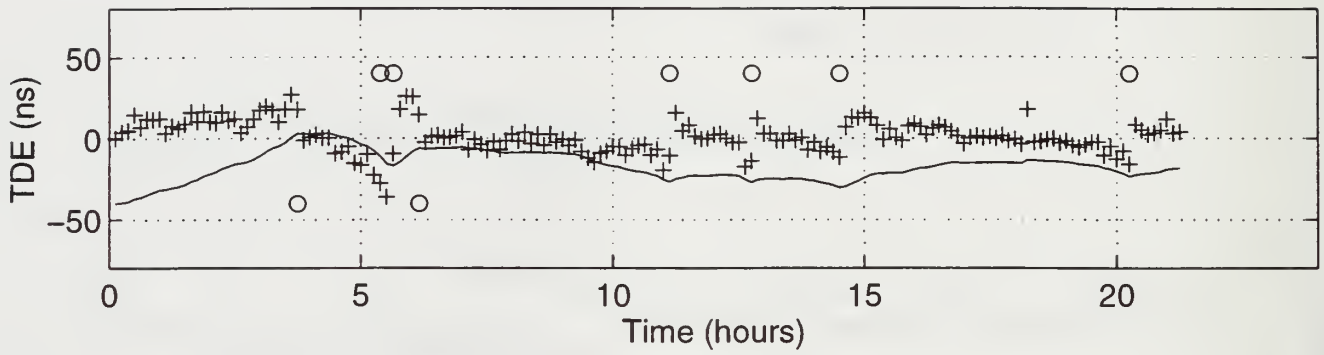
(a) SEUS Station Yankee



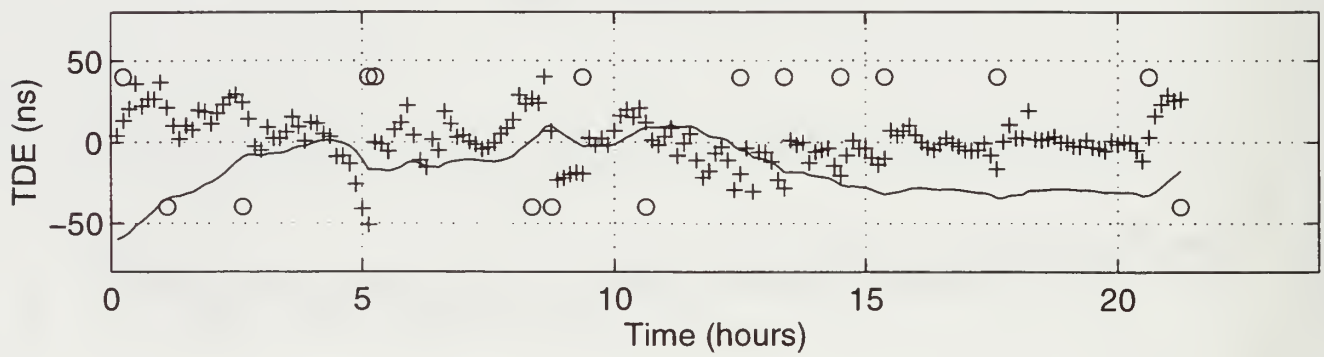
(b) SEUS Station Zulu

Figure 45. Kalman Filter Strip Chart for South East United States (SEUS) Loran-C Chain recorded on 18 January 1997: o Linear Phase Adjustment (LPA) (magnitude times 2); + Time Difference Error (TDE); — Cumulative TDE.

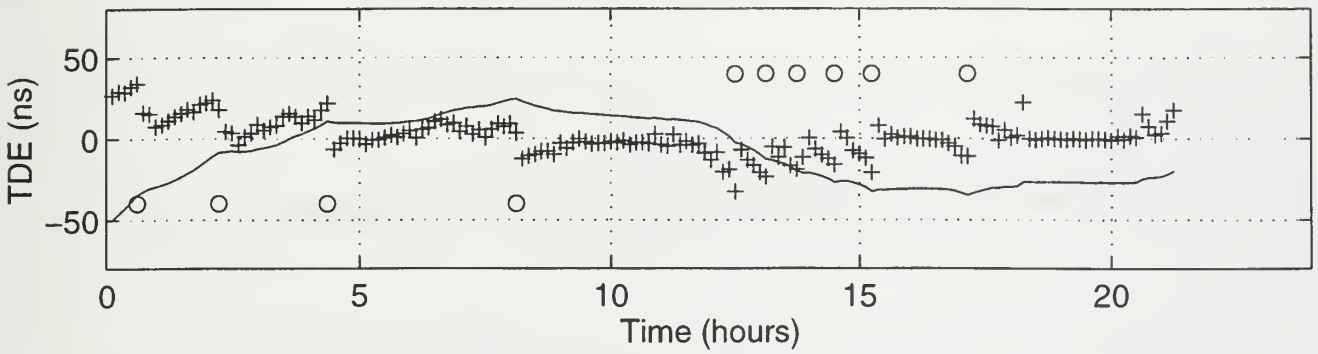
b. 20 January 1997



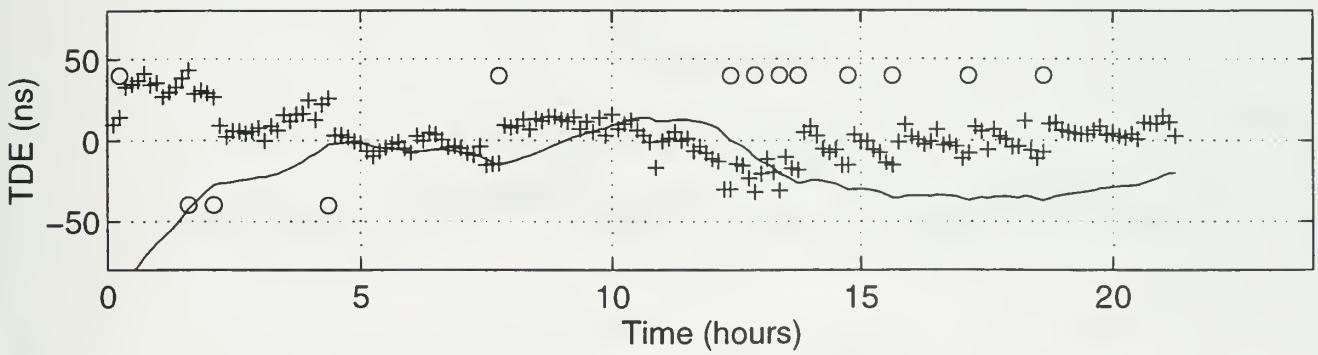
(a) SEUS Station Whiskey



(b) SEUS Station Xray



(c) SEUS Station Yankee

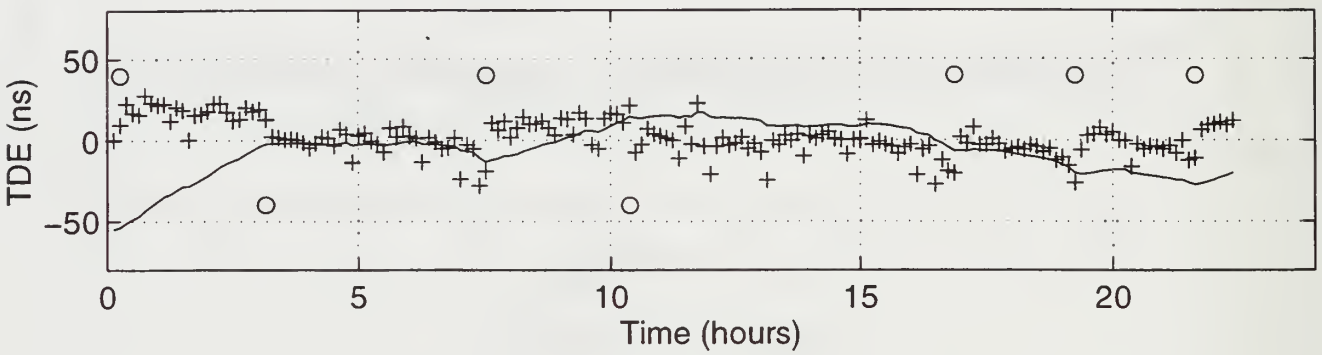


(d) SEUS Station Zulu

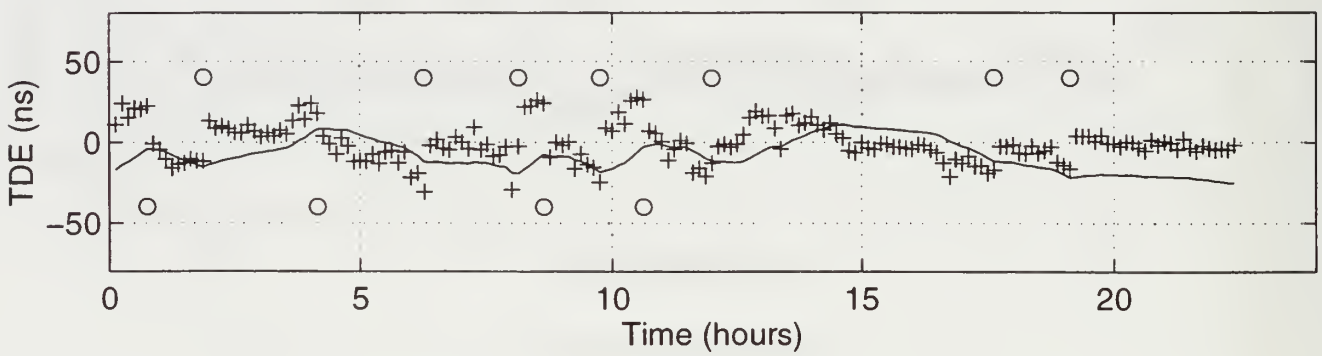
Figure 46. Kalman Filter Strip Chart for South East United States (SEUS) Loran-C Chain recorded on 20 January 1997: o Linear Phase Adjustment (LPA) (magnitude times 2); + Time Difference Error (TDE); — Cumulative TDE.

2. SOUTH CENTRAL UNITED STATES (SOCUS)

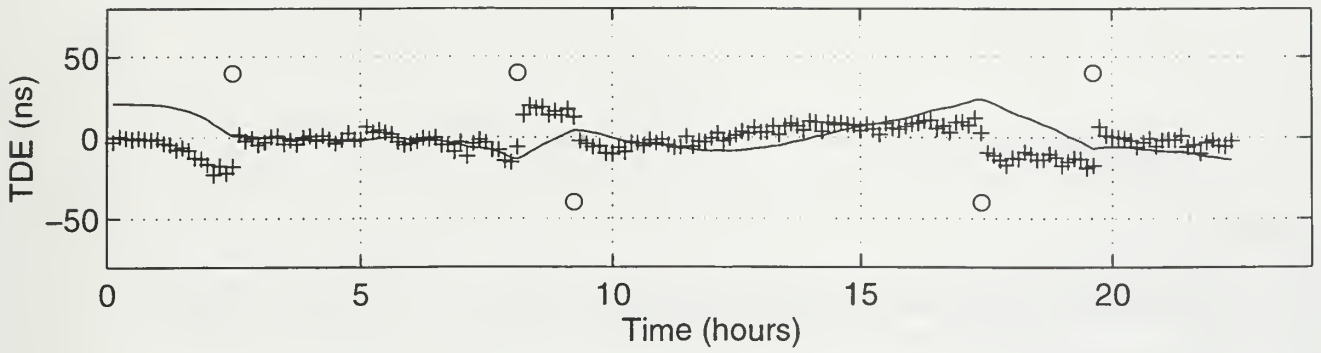
a. 18 January 1997



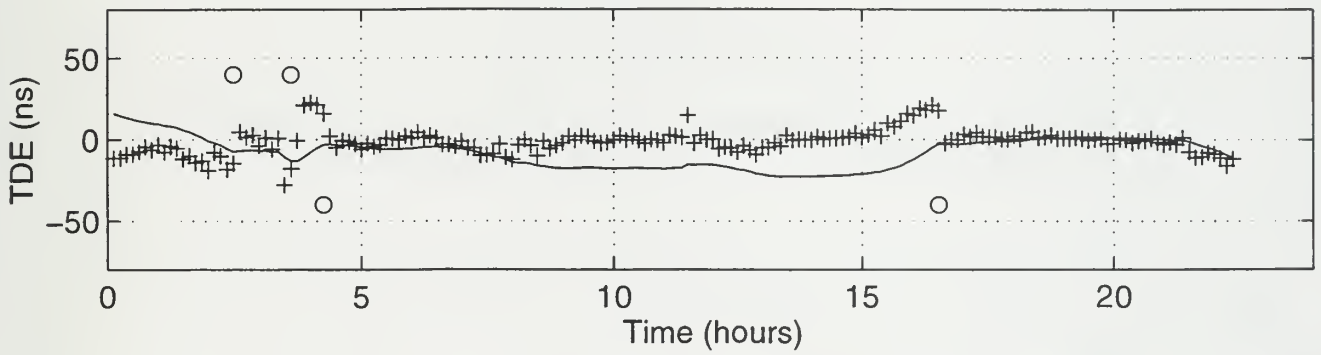
(a) SOCUS Station Victor



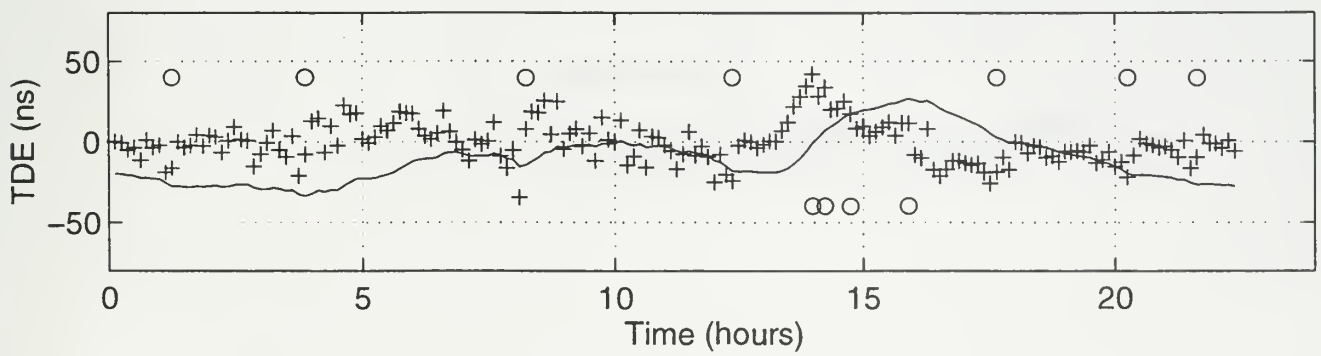
(b) SOCUS Station Whiskey



(c) SOCUS Station Xray



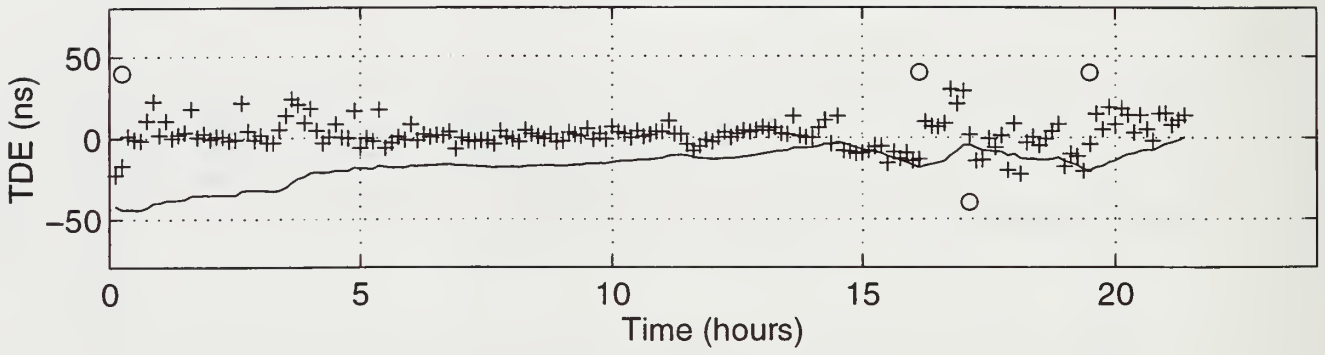
(d) SOCUS Station Yankee



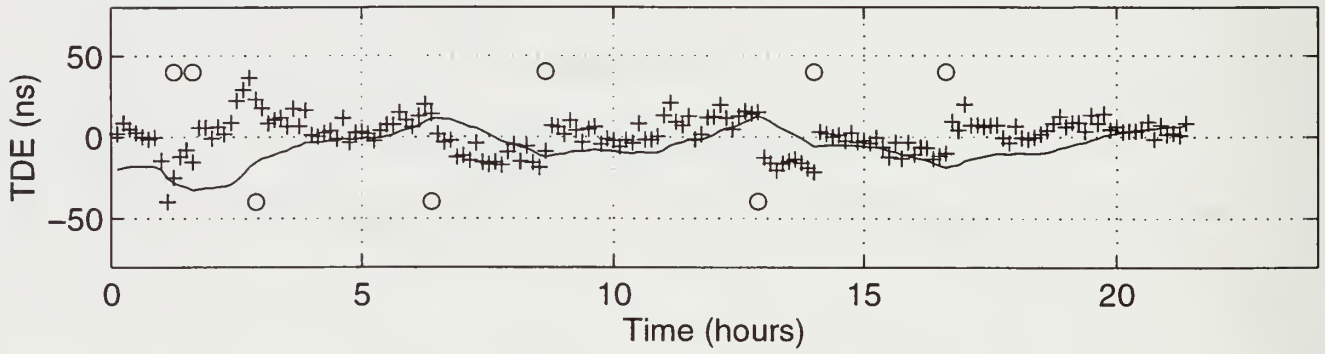
(e) SOCUS Station Zulu

Figure 47. Kalman Filter Strip Chart for South Central United States (SOCUS) Loran-C Chain recorded on 18 January 1997: o Linear Phase Adjustment (LPA) (magnitude times 2); + Time Difference Error (TDE); — Cumulative TDE.

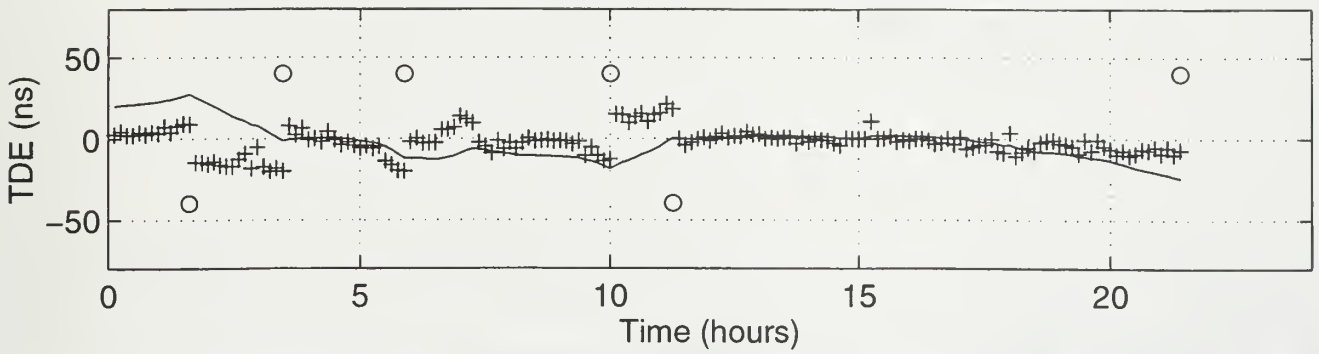
b. 20 January 1997



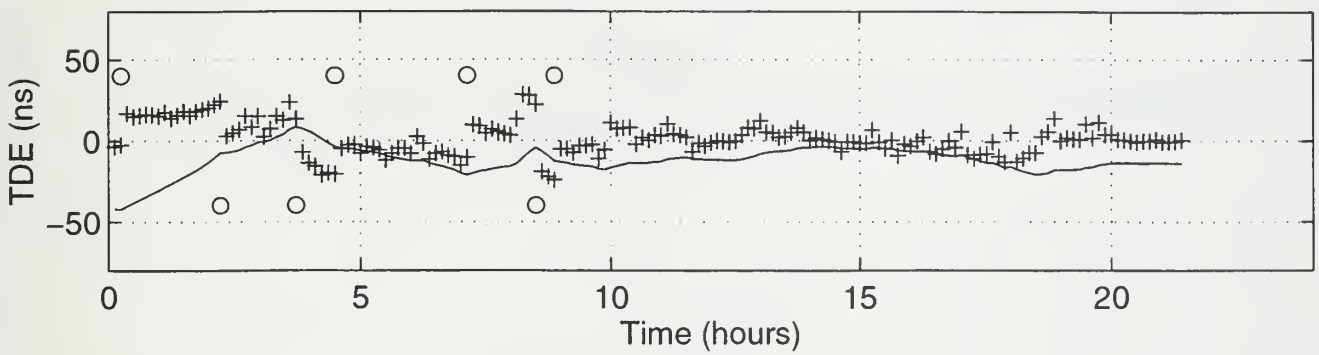
(a) SOCUS Station Victor



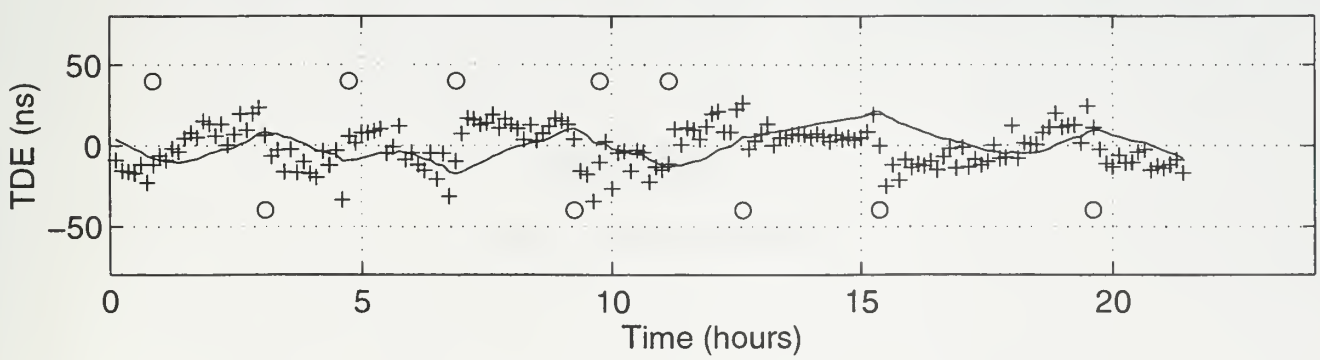
(b) SOCUS Station Whiskey



(c) SOCUS Station Xray



(d) SOCUS Station Yankee

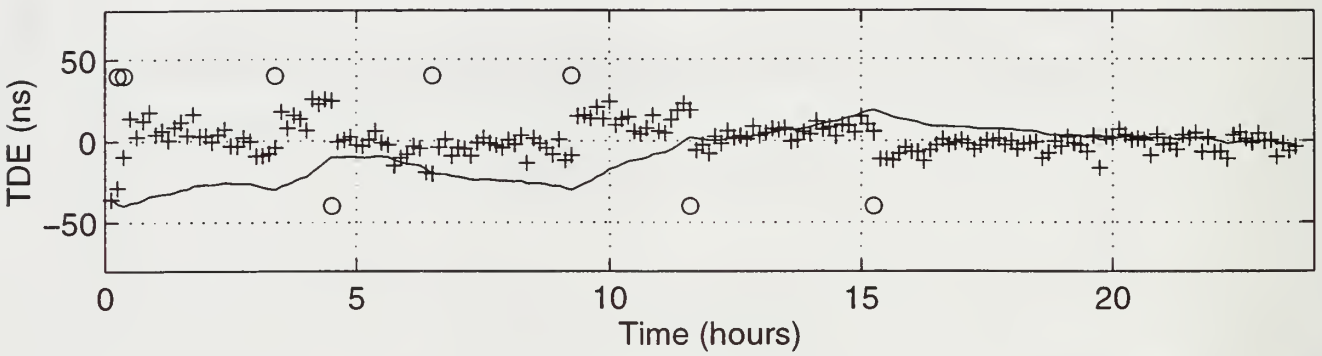


(e) SOCUS Station Zulu

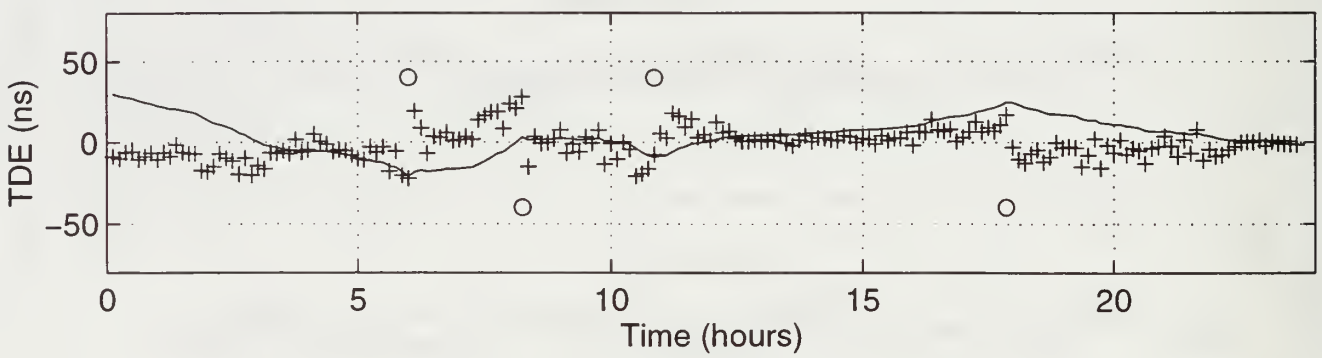
Figure 48. Kalman Filter Strip Chart for South Central United States (SOCUS) Loran-C Chain recorded on 20 January 1997: o Linear Phase Adjustment (LPA) (magnitude times 2); + Time Difference Error (TDE); — Cumulative TDE.

3. NORTH CENTRAL UNITED STATES (NOCUS)

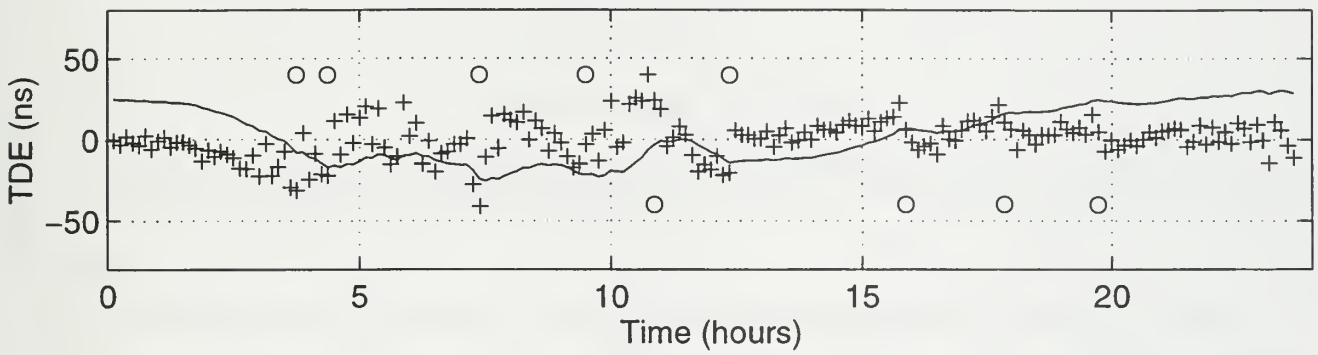
a. 4 May 1997



(a) NOCUS Station Whiskey



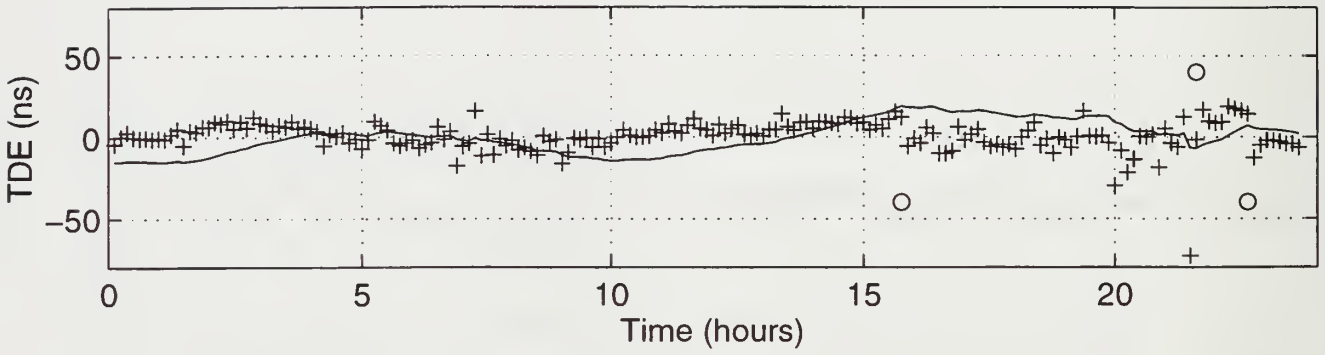
(b) NOCUS Station Xray



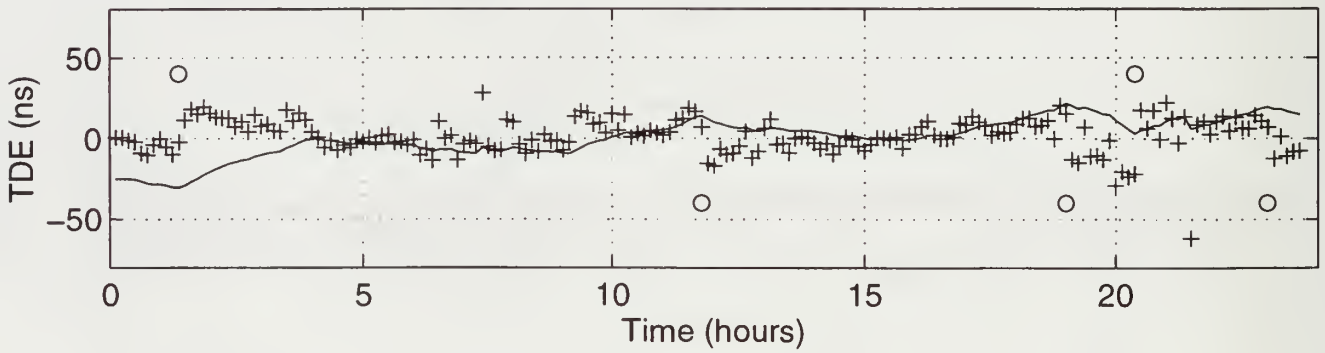
(c) NOCUS Station Yankee

Figure 49. Kalman Filter Strip Chart for North Central United States (NOCUS) Loran-C Chain recorded on 4 May 1997: o Linear Phase Adjustment (LPA) (magnitude times 2); + Time Difference Error (TDE); — Cumulative TDE.

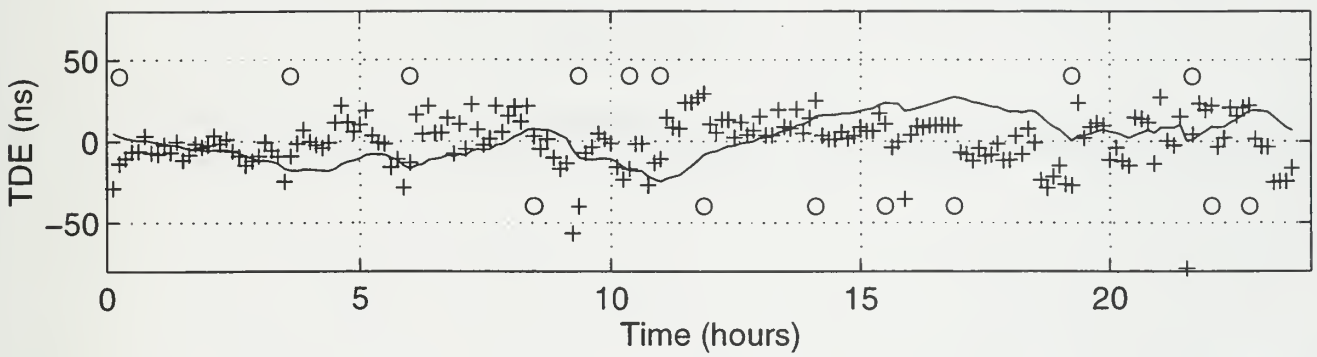
b. 5 May 1997



(a) NOCUS Station Whiskey



(b) NOCUS Station Xray

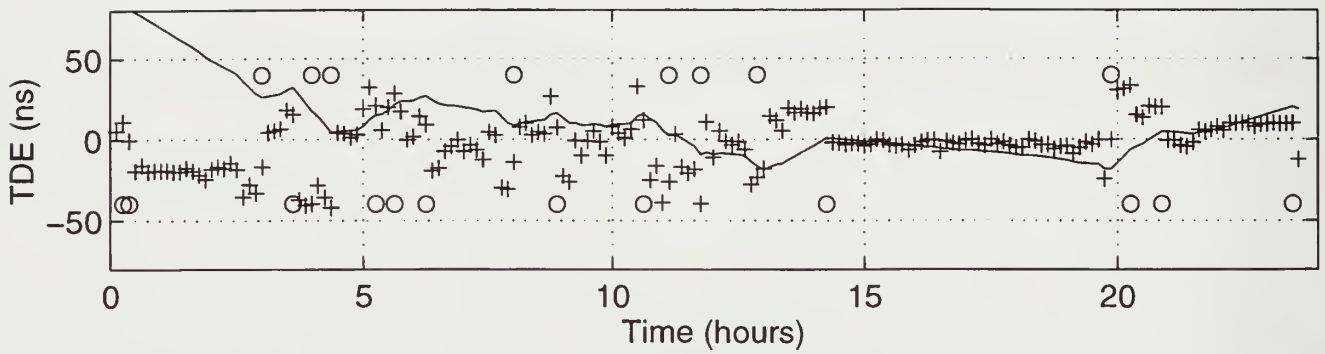


(c) NOCUS Station Yankee

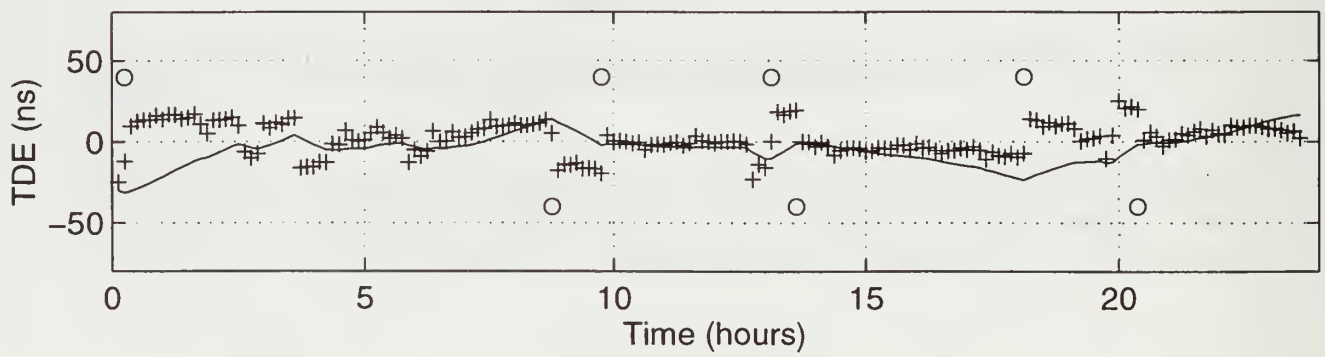
Figure 50. Kalman Filter Strip Chart for North Central United States (NOCUS) Loran-C Chain recorded on 5 May 1997: o Linear Phase Adjustment (LPA) (magnitude times 2); + Time Difference Error (TDE); — Cumulative TDE.

4. UNITED STATES WEST COAST (USWC)

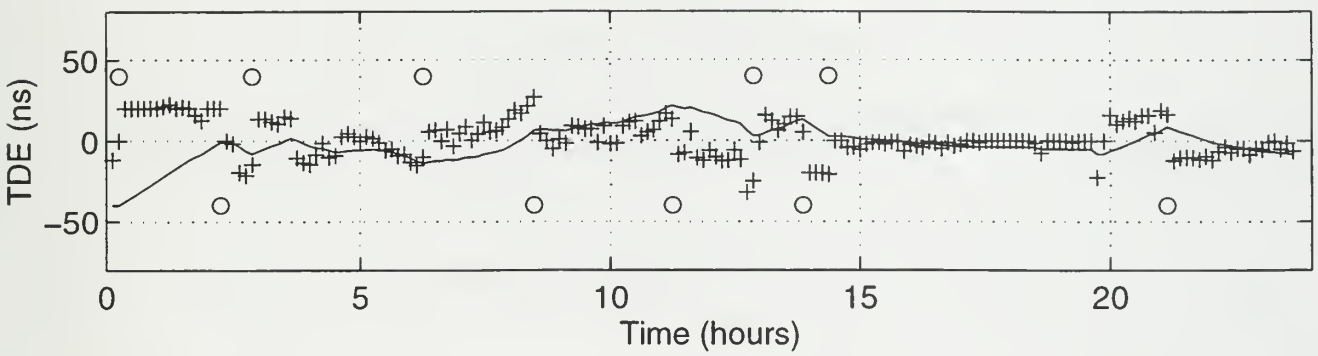
a. 4 May 1997



(a) USWC Station Whiskey



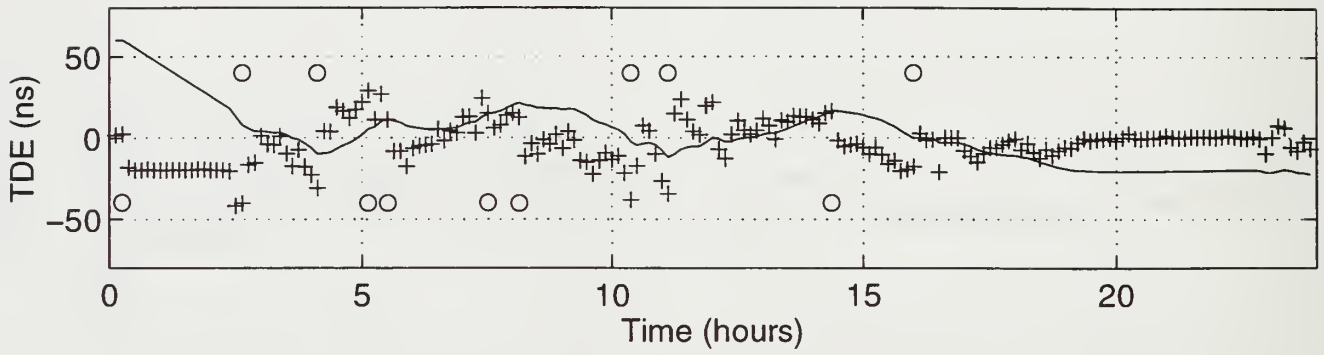
(b) USWC Station Xray



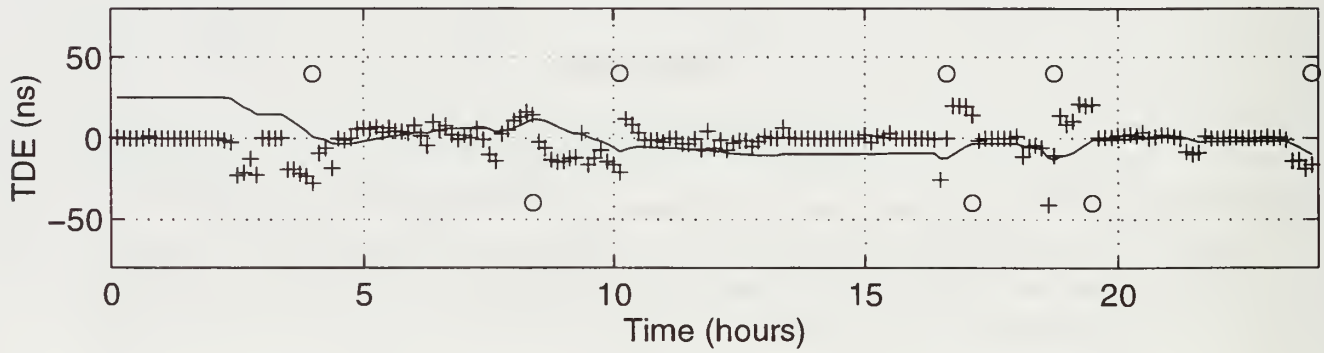
(c) USWC Station Yankee

Figure 51. Kalman Filter Strip Chart for United States West Coast (USWC) Loran-C Chain recorded on 4 May 1997: o Linear Phase Adjustment (LPA) (magnitude times 2); + Time Difference Error (TDE); — Cumulative TDE.

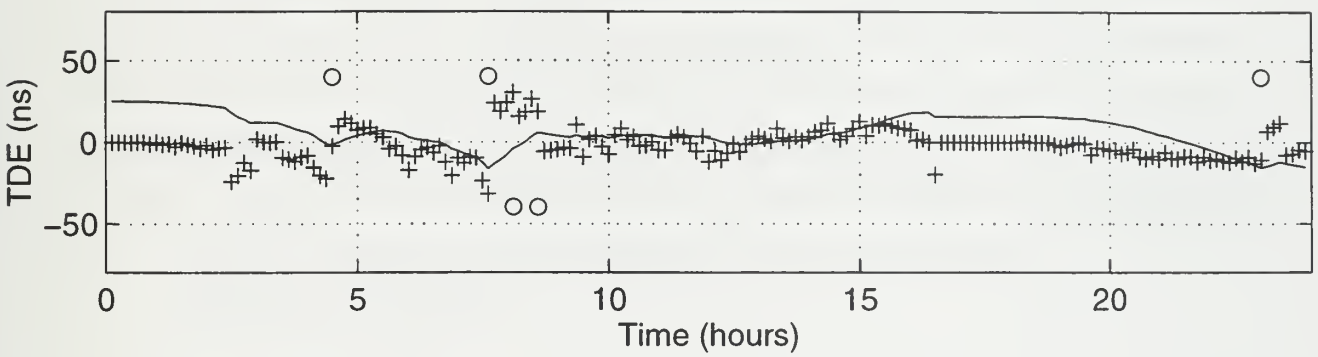
b. 5 May 1997



(a) USWC Station Whiskey



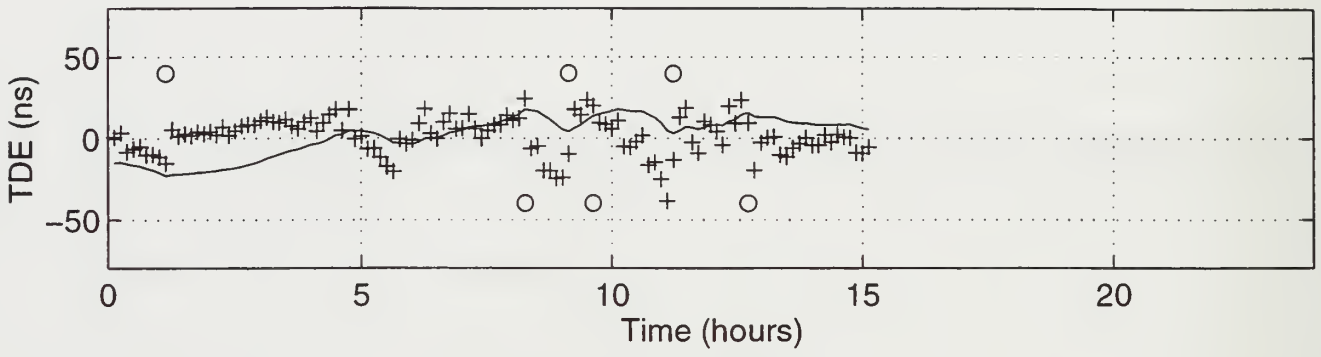
(b) USWC Station Xray



(c) USWC Station Yankee

Figure 52. Kalman Filter Strip Chart for United States West Coast (USWC) Loran-C Chain recorded on 5 May 1997: o Linear Phase Adjustment (LPA) (magnitude times 2); + Time Difference Error (TDE); — Cumulative TDE.

5. KODIAK



(a) Kodiak Station Zulu

Figure 53. Kalman Filter Strip Chart for Kodiak Loran-C Chain: o Linear Phase Adjustment (LPA) (magnitude times 2); + Time Difference Error (TDE); — Cumulative TDE.

APPENDIX D. MATLAB CODE

This appendix contains seven sections of Matlab code, and associated function and variable descriptions. The first section contains a listing of the data sets and major functions with a brief description of their purpose. Also contained in Section 1 are the function variables along with their definition and functions in which they are used. Sections 3 and 4 contain the functions *pid* and *kalman*, respectively, as well as their associated driver functions. Driver functions are script files used to make repeated calls to a function using numerous different input options without having to duplicate the basic function. Section 5 contains the Matlab code to preprocess the recorded data, and Section 6 contains the code used to generate the CALOC strip charts based on the recorded data. Finally, the code used to generate the plot titles is contained in Section 7.

1. DATA SETS, FUNCTION AND VARIABLE DESCRIPTIONS

a. Data Sets

kodiak.dat — North Pacific (NORPAC) Station Zulu recorded on 11 September 1996.

nocus4my.dat — North Central United States (NOCUS) Stations Whiskey, Xray and Yankee recorded on 4 May 1997.

nocus5my.dat — North Central United States (NOCUS) Stations Whiskey, Xray and Yankee recorded on 5 May 1997.

seus18jn.dat — South East United States (SEUS) Stations Yankee and Zulu recorded on 18 January 1997.

seus20jn.dat — South East United States (SEUS) Stations Whiskey, Xray, Yankee and Zulu recorded on 20 January 1997.

socus18j.dat South Central United States (SOCUS) Stations Victor, Whiskey, Xray, Yankee and Zulu recorded on 18 January 1997.

socus20j.dat South Central United States (SOCUS) Stations Victor, Whiskey, Xray, Yankee and Zulu recorded on 20 January 1997.

uswc4my.dat South East United States (SEUS) Stations Whiskey, Xray and Yankee recorded on 4 May 1997.

uswc5my.dat South East United States (SEUS) Stations Whiskey, Xray and Yankee recorded on 5 May 1997.

b. Functions

firham.m This function returns the difference equation coefficients for a linear phase FIR filter multiplied by a Hamming window of length M and digital cut-off frequency, ω_c.

kalman.m This function implements a Kalman Filter to control the Time Difference Error (TDE_data) of the LORAN-C System. The two controlled parameters are the cumulative TDE_data (long-term error) and the current TDE_data (short-term error). The estimate for the current TDE_data is generated using a Kalman estimator, which is then used as one of the feedback parameters for the Kalman filter. The cumulative TDE_data is computed by adding the number of minutes divided by 60 times the current TDE_data to account for the number of averages per hour.

parse_title.m This function parses the data file name string and generates a string for the title of the output graph.

pid.m — This function implements a PID controller to control the Time Difference Error (TDE) of the LORAN-C System. The two controlled parameters are the cumulative TDE (long-term error) and the current TDE (short-term error). The cumulative TDE is computed by adding the number of minutes divided by 60 times the current TDE to account for the number of averages per hour.

c. Data Vector Variables

seus18jnY	Data vector for SEUS Yankee on 18 January 1997.
seus18jnZ	Data vector for SEUS Zulu on 18 January 1997.
seus20jnW	Data vector for SEUS Victor on 20 January 1997.
seus20jnX	Data vector for SEUS Whiskey on 20 January 1997.
seus20jnY	Data vector for SEUS Yankee on 20 January 1997.
seus20jnZ	Data vector for SEUS Zulu on 20 January 1997.
socus18jV	Data vector for SOCUS Victor on 18 January 1997.
socus18jW	Data vector for SOCUS Whiskey on 18 January 1997.
socus18jX	Data vector for SOCUS Xray on 18 January 1997.
socus18jY	Data vector for SOCUS Yankee on 18 January 1997.
socus18jZ	Data vector for SOCUS Zulu on 18 January 1997.
socus20jV	Data vector for SOCUS Victor on 20 January 1997.
socus20jW	Data vector for SOCUS Whiskey on 20 January 1997.
socus20jX	Data vector for SOCUS Xray on 20 January 1997.
socus20jY	Data vector for SOCUS Yankee on 20 January 1997.
socus20jZ	Data vector for SOCUS Zulu on 20 January 1997.
uswc4myW	Data vector for USWC Whiskey on 4 May 1997.
uswc4myX	Data vector for USWC Xray on 4 May 1997.

uswc4myY	Data vector for USWC Yankee on 4 May 1997.
uswc5myW	Data vector for USWC Whiskey on 5 May 1997.
uswc5myX	Data vector for USWC Xray on 5 May 1997.
uswc5myY	Data vector for USWC Yankee on 5 May 1997.

nocus4myW	Data vector for NOCUS Whiskey on 4 May 1997
nocus4myX	Data vector for NOCUS Xray on 4 May 1997
nocus4myY	Data vector for NOCUS Yankee on 4 May 1997
nocus5myW	Data vector for NOCUS Whiskey on 5 May 1997
nocus5myX	Data vector for NOCUS Xray on 5 May 1997
nocus5myY	Data vector for NOCUS Yankee on 5 May 1997

kodiakZ	Data vector for NORPAC Zulu on 11 September 1996

d. Other Variables

A	Weight in the Matlab function DLQR constraint equation associated with the feedback parameters.	kalman
B	Weight in the Matlab function DLQR constraint equation associated with the LPA.	kalman
conv_data	Result of the convolution between the FIR filter and data vector.	kalman, pid
corr20	Constant value used for applying corrections to the data to remove CALOC influence due to a 20 ns LPA.	Data Correction

corr40	Constant value used for applying corrections to the data to remove CALOC influence due to a 40 ns LPA.	Data Correction
cum_TDE	Cumulative time difference error.	kalman, pid
data_cols	Number of data points in the vector after decimation by factor of time_int.	kalman, pid
dec_data	Reshaped result of convolution in preparation for decimation.	kalman, pid
delay	Constant value of actual delay experienced between an LPA being ordered and its effect being seen at the monitor station.	Data Correction
fir_filter	Difference equation coefficients for the linear phase FIR filter with Hamming window.	kalman, pid
graph_title	String used to generate title for plot of strip chart.	kalman, pid
Kp	Proportional weight for PID Controller.	pid
Ki	Integral weight for PID Controller.	pid
Kd	Derivative weight for PID Controller.	pid
L	Control vector used to weight feedback parameters TDE_est and cum_TDE.	kalman
lpa	Locations of LPAs used for plotting the CALOC Strip Charts.	kalman, pid

LPA	Linear phase adjustment that is quantizer output.	kalman, pid
LPA_int	Constant value of the linear phase adjustment increment.	kalman, pid
minutes	Number of minutes between TDE averages used to determine decimation interval.	kalman, pid
num_cols	Number of columns in data set.	kalman, pid
num_rows	Number of rows in data set.	kalman, pid
out	Output of PID Controller or Kalman Estimator as input to quantizer.	kalman, pid
p	Correlation matrix of the state error equation.	kalman
plotlpa	Vector of NaN used to generate LPA vector by indexing on quantizer output.	kalman, pid
Q	Covariance of the system noise term associated with the LPA in the Kalman model.	kalman
Q_lqr	Weight in the Matlab function DLQR cost function associated with the feedback parameters.	kalman
quant_var	Half-step size for LPA quantizer.	kalman, pid
R	Covariance of the sensor noise term associated with the TDE in the Kalman model.	kalman
R_lqr	Weight in the Matlab function DLQR cost function associated with the LPA.	kalman

TDE_data	Data decimated by factor of time_int.	kalman, pid
TDE_est	Time difference estimate for Kalman filter.	kalman
time_int	Number of data points that make up a decimation interval.	kalman, pid
xaxis	Vector used to plot output values to common reference in time.	kalman, pid

2. LINEAR PHASE FIR FILTER WITH HAMMING WINDOW

```
================================================================
================================================================

function h = firham(M, omega_c)
% function h = firham(M, omega_c)
%
% FIRHAM Returns the difference equation coefficients for a linear
%        phase FIR filter multiplied by a Hamming window of length
%        M and digital cutoff frequency, omega_c.

% Initialize variables for filter length and coefficients
filter_length = 0:M-1;
diffeq_coeff = zeros(size(filter_length));

% Generate difference equation coefficients of linear phase FIR
% filter using equation (8.1.20) on page 584 from Proakis
% Digital Signal Processing
diffeq_coeff = sin(omega_c * (filter_length - (M-1)/2)) ./ ...
   (pi * (filter_length - (M-1)/2));

% If filter length is odd, then h(n) at (M-1)/2 is omega_c/pi
if rem(M,2)==1
   diffeq_coeff((M-1)/2 + 1) = omega_c/pi;
end

% Generate Hamming window of length M
ham_M = hamming(M);

% Multiply linear phase FIR filter by Hamming window of length M
fir_ham = diffeq_coeff .* ham_M';

% Make filter unit gain
h = fir_ham./sum(fir_ham);
```

3. PID CONTROLLER FUNCTION AND DRIVERS

a. Function

```
========================================================================
========================================================================

function out = pid(data_set, init_cond, graph_title)
% LORAN-C Thesis
%
% PID  This function implements a PID controller to control the Time
%      time Difference Error (TDE) of the LORAN-C System.  The two
%      controlled parameters are the cumulative TDE (long-term error)
%      and the current TDE (short-term error).  The cumulative TDE is
%      computed by adding the number of minutes divided by 60 times
%      the current TDE to account for the number of averages per hour.
%
% The inputs to the function are an integer data vector, the initial
% condition in integer form, and a string that will be the title of
% of the graph output.  The minimum input required is a valid data
% vector.  The initial condition or graph title may be omitted.  If
% either areomitted, their default values will be used.  The default
% values for the other inputs are:
%      init_cond = 0;
%      graph_title = '';

% Function call error checking.

% Verify that function call has 1, 2 or 3 inputs.
if nargin > 3
   error('There are too many arguments in the function call.')
elseif nargin == 0
   error('Invalid function call.  At a minimum, function call must
have a data vector.')
end

% Verify first entry is a valid integer data vector.
if isstr(data_set)
   error('Data set has been entered as a string.  Data set must be a
vector.')
end
```

```
% Verify each input is of the correct form.
%
%
if nargin == 3
   if isstr(init_cond)
      error('Initial condition entered as a string - must be an
integer.')
   else
      mbint(init_cond)
   end
   if ~isstr(graph_title)
      error('Graph title is not a string.')
   end
elseif nargin == 2
   if (isstr(init_cond))
      graph_title = init_cond;
      init_cond = 0;
      disp('   Assuming second input to function is graph')
      disp('   title.  Setting initial condition to 0.')
      pause(5)
   else
      graph_title = '';
   end
else
   init_cond = 0;
   graph_title = '';
end

% Generate linear phase FIR filter with Hamming window
%    FIRHAM   Returns the difference equation coefficients for a
%             linear phase FIR filter multiplied by a Hamming window
%             of length M and digital cutoff frequency, omega_c.
fir_filter = firham(32, pi/45);

% Determine the size of the data file.
[num_rows, num_cols] = size(data_set);

% Constants for decimation.
%
% Determine number of data points after decimation.  The length
% of time equating to the decimation interval is determined by the
% variable, minutes.  Since each data point represents 10 seconds in
```

```
% real time, there are (minutes x 6) data points in the decimation
% interval.  In this particular case, there are 45 data points in
% each 7.5 minute interval.  Using the Matlab command floor rounds
% down the result of the operation to get an integer value used to
% reshape the data for decimation.
minutes = 7.5;
time_int = minutes * 6;
data_cols = floor(num_rows/time_int);

% Convolve FIR filter with data.
conv_data = conv(fir_filter, data_set);

% Reshape data for decimation.
dec_data = reshape(conv_data(1:(time_int*data_cols)), time_int,
                   data_cols);

% Decimate by factor of time_int to obtain a vector of TDEs taken
% at a 7.5 minute interval.  These values are the input to the control
% algorithm.
TDE_data = dec_data(time_int, :);

% Initialize variables
LPA_int = 20;
quant_var = 10;
Kp = -.35;
Ki = -.12;
Kd = -.2;
out = zeros(size(TDE_data));
LPA = zeros(size(TDE_data));
cum_TDE = zeros(size(TDE_data));
cum_TDE(1) = init_cond;

% PID Controller
%
% This for loop implements a PID controller.  The loop starting value
% of 2 is a constraint of Matlab requiring the index of a vector to
% be a positive, non-zero integer.

for time = 2:data_cols

   % Calculate the cumulative time difference error (cum_TDE) by
   % adding the current TDE (TDE_data) to the previous value for
```

```
% the cumulative TDE.  The current TDE is scaled by a factor of
% minutes/60 to account for the number of corrections per hour.
cum_TDE(time) = cum_TDE(time-1) + TDE_data(time) * (minutes/60);

% Output of the PID controller prior to quantization.  The integral
% weight, Ki, is applied to the cumulative TDE; the proportional
% weight, Kp, is applied to the current TDE; and the derivative
% weight, Kd, is applied to the difference between the current TDE
% and the value for the current TDE from the previous time step.
out(time) = (Ki * cum_TDE(time)) + (Kp * TDE_data(time)) + ...
   (Kd * (TDE_data(time) - TDE_data(time - 1)));

% LPA Decision
%
% The output of the PID controller is quantized to interface
% with the Loran-C transmitter timing controller which accepts
% corrections in 20 ns intervals.  It is a mid-tread quantizer
% with equal sized steps of 20 ns.  The step size may be varied
% by adjusting the quantization variable, quant_var.  The output
% of the quantizer is the linear phase adjustment (LPA).
if out(time) >= 17*quant_var
   LPA(time) = 180;
elseif out(time) >= 15*quant_var
   LPA(time) = 160;
elseif out(time) >= 13*quant_var
   LPA(time) = 140;
elseif out(time) >= 11*quant_var
   LPA(time) = 120;
elseif out(time) >= 9*quant_var
   LPA(time) = 100;
elseif out(time) >= 7*quant_var
   LPA(time) = 80;
elseif out(time) >= 5*quant_var
   LPA(time) = 60;
elseif out(time) >= 3*quant_var
   LPA(time) = 40;
elseif out(time) >= quant_var
   LPA(time) = 20;
elseif out(time) >= -quant_var
   LPA(time) = 0;
elseif out(time) >= -3*quant_var
   LPA(time) = -20;
```

```
   elseif out(time) >= -5*quant_var
      LPA(time) = -40;
   elseif out(time) >= -7*quant_var
      LPA(time) = -60;
   elseif out(time) >= -9*quant_var
      LPA(time) = -80;
   elseif out(time) >= -11*quant_var
      LPA(time) = -100;
   elseif out(time) >= -13*quant_var
      LPA(time) = -120;
   elseif out(time) >= -15*quant_var
      LPA(time) = -140;
   elseif out(time) >= -17*quant_var
      LPA(time) = -160;
   elseif out(time) >= -19*quant_var
      LPA(time) = -180;
   end

   % Apply LPA to the remaining data set.  This is a function of
   % non-real time implementation.  If implemented in real-time,
   % the LPA would be applied to the transmitter and the shift in
   % TDE would be seen in the data points.
   TDE_data(time:data_cols) = TDE_data(time:data_cols) + LPA(time);

end

% Generate vector to plot LPA.  Dividing by zero generates a
% vector of NaN so that only the LPA locations determined in
% the for loop below will be plotted.
plotlpa = zeros(size(LPA))/0.0;

% This for loop determines the locations of the LPAs by indexing
% the capability of Matlab to index a vector based on its non-zero
% values.  The location of each LPA is transferred to the NaN
% vector generated above.  Since Matlab does not plot NaN, the only
% values that are plotted are the LPA locations.
for index=1:data_cols
   if LPA(index)~=0
      plotlpa(index)=LPA(index);
   end
end
```

```
% Generate the LORAN Strip Charts
%
% The plots are generated using subplot so that they appear with
% similar resolution to the existing strip charts.  The plots
% include the TDE averages (TDE_data), the cumulative TDE, and the
% locations of the LPAs input by CALOC.  The LPAs are plotted
% with a multiplicative factor of 2 solely for ease of display.
% The variable xaxis is used for common reference when plotting.
xaxis = [1:num_rows/time_int]/(60/minutes);

figure
subplot(311), plot(xaxis, TDE_data, 'w+', xaxis, cum_TDE(1:data_cols),
                   'w-', xaxis, plotlpa*2, 'wo')
xlabel('Time (hours)'), ylabel('TDE (ns)')
title(graph_title)
axis([0 24 -80 80]), grid on
```

b. Drivers

```
========================================================================
========================================================================

data18seuscorr

graph_title = parse_title('seus_18jnY');
pid(seus18jnY, -31, graph_title);

graph_title = parse_title('seus_18jnZ');
pid(seus18jnZ, -77, graph_title);



data20seuscorr

graph_title = parse_title('seus_20jnW');
pid(seus20jnW, -40, graph_title);

graph_title = parse_title('seus_20jnX');
pid(seus20jnX, -60, graph_title);

graph_title = parse_title('seus_20jnY');
```

```
pid(seus20jnY, -50, graph_title);

graph_title = parse_title('seus_20jnZ');
pid(seus20jnZ, -92, graph_title);



data18socuscorr

graph_title = parse_title('socus18jnV');
pid(socus18jV, -55, graph_title);

graph_title = parse_title('socus18jnW');
pid(socus18jW, -17, graph_title);

graph_title = parse_title('socus18jnX');
pid(socus18jX, 21, graph_title);

graph_title = parse_title('socus18jnY');
pid(socus18jY, 16, graph_title);

graph_title = parse_title('socus18jnZ');
pid(socus18jZ, -20, graph_title);



data20socuscorr

graph_title = parse_title('socus20jnV');
pid(socus20jV, -42, graph_title);

graph_title = parse_title('socus20jnW');
pid(socus20jW, -20, graph_title);

graph_title = parse_title('socus20jnX');
pid(socus20jX, 20, graph_title);

graph_title = parse_title('socus20jnY');
pid(socus20jY, -42, graph_title);

graph_title = parse_title('socus20jnZ');
pid(socus20jZ, 4, graph_title);
```

```
data4nocuscorr

graph_title = parse_title('nocus04myW');
pid(nocus4myW, -35, graph_title);

graph_title = parse_title('nocus04myX');
pid(nocus4myX, 30, graph_title);

graph_title = parse_title('nocus04myY');
pid(nocus4myY, 25, graph_title);



data5nocuscorr

graph_title = parse_title('nocus05myW');
pid(nocus5myW, -15, graph_title);

graph_title = parse_title('nocus05myX');
pid(nocus5myX, -25, graph_title);

graph_title = parse_title('nocus05myY');
pid(nocus5myY, 5, graph_title);



data4uswccorr

graph_title = parse_title('uswc_04myW');
pid(uswc4myW, 80, graph_title);

graph_title = parse_title('uswc_04myX');
pid(uswc4myX, -30, graph_title);

graph_title = parse_title('uswc_04myY');
pid(uswc4myY, -40, graph_title);
```

```
data5uswccorr

graph_title = parse_title('uswc_05myW');
pid(uswc5myW, 60, graph_title);

graph_title = parse_title('uswc_05myX');
pid(uswc5myX, 25, graph_title);

graph_title = parse_title('uswc_05myY');
pid(uswc5myY, 25, graph_title);



datakodiakcorr
graph_title = parse_title('kodik11spZ');
pid(kodiakZ, -15, graph_title);
```

4. KALMAN FILTER FUNCTION AND DRIVERS

a. Function

```
%========================================================================
%========================================================================

function out = kalman(data_set, init_cond, graph_title)
% LORAN-C Thesis     .
%
% KALMAN  This function implements a Kalman Filter to control the
%         Time Difference Error (TDE_data) of the LORAN-C System.
%         The two controlled parameters are the cumulative TDE_data
%         (long-term error) and the current TDE_data (short-term
%         error).  The estimate for the current TDE_data is generated
%         using a Kalman estimator, which is then used as one of the
%         feedback parameters for the Kalman filter.  The cumulative
%         TDE_data is computed by adding the number of minutes divided
%         by 60 times the current TDE_data to account for the number
%         of averages per hour.

% Function call error checking.

% Verify that function call has 1, 2 or 3 inputs.
if nargin > 3
   error('There are too many arguments in the function call.')
elseif nargin == 0
   error('Invalid function call.  At a minimum, function call must
have a data vector.')
end

% Verify first entry is a valid integer data vector.
if isstr(data_set)
   error('Data set has been entered as a string.  Data set must be a
vector.')
end

% Verify each input is of the correct form.
%
%
if nargin == 3
   if isstr(init_cond)
```

```
      error('Initial condition entered as a string - must be an
integer.')
   else
      mbint(init_cond)
   end
   if ~isstr(graph_title)
      error('Graph title is not a string.')
   end
elseif nargin == 2
   if (isstr(init_cond))
      graph_title = init_cond;
      init_cond = 0;
      disp('   Assuming second input to function is graph')
      disp('   title.  Setting initial condition to 0.')
      pause(5)
   else
      graph_title = '';
   end
else
   init_cond = 0;
   graph_title = '';
end

% Generate linear phase FIR filter with Hamming window
%    FIRHAM   Returns the difference equation coefficients for a
%             linear phase FIR filter multiplied by a Hamming window
%             of length M and digital cutoff frequency, omega_c.
fir_filter = firham(32, pi/45);

% Determine the size of the data file.
[num_rows, num_cols] = size(data_set);

% Constants for decimation.
%
% Determine number of data points after decimation.  The length
% of time equating to the decimation interval is determined by the
% variable, minutes.  Since each data point represents 10 seconds in
% real time, there are (minutes x 6) data points in the decimation
% interval.  In this particular case, there are 45 data points in
% each 7.5 minute interval.  Using the Matlab command floor rounds
% down the result of the operation to get an integer value used to
% reshape the data for decimation.
```

```
minutes = 7.5;
time_int = minutes * 6;
data_cols = floor(num_rows/time_int);

% Convolve FIR filter with data.
conv_data = conv(fir_filter, data_set);

% Reshape data for decimation.
dec_data = reshape(conv_data(1:(time_int*data_cols)), time_int,
                   data_cols);

% Decimate by factor of time_int to obtain a vector of TDE_datas taken
% at a 7.5 minute interval.  These values are the input to the control
% algorithm.
TDE_data = dec_data(time_int, :);

% Initialize variables
LPA_int = 20;
quant_var = 25;
R = median(std(dec_data));
Q = .5;
p = zeros(size(TDE_data));
p(1) = 10;
LPA = zeros(size(TDE_data));
cum_TDE = zeros(size(TDE_data));
cum_TDE(1) = init_cond;
TDE_est = zeros(size(TDE_data));

% Linear Quadratic Regulator
%    L = DLQR(A, B, Q, R) calculates the optimal feedback gain matrix
%                         K such that the feedback law
%                            u[n] = -Kx[n]
%                         minimizes the cost function
%                            J = Sum {x'Qx + u'Ru}
%                         subject to the constraint equation:
%                            x[n+1] = Ax[n] + Bu[n]
A = [1 0; 1 1];
B = [1; 0];
R_lqr = 1;
Q_lqr = diag([5 5]);
L = dlqr(A, B, Q_lqr, R_lqr);
```

```
% Kalman estimator
for time = 2:data_cols

   % Apply LPA to the remaining data set.  This is a function of
   % non-real time implementation.  If implemented in real-time,
   % the LPA would be applied to the transmitter and the shift in
   % TDE would be seen in the data points.
   TDE_data(time:data_cols) = TDE_data(time:data_cols) + LPA(time-1);

   % Estimator for the current time difference error (TDE_est).
   TDE_est(time) = TDE_est(time-1) + LPA(time-1) + ...
      (p(time-1)/(R + p(time-1))) * (TDE_data(time-1) - ...
       TDE_est(time-1));
   p(time) = p(time-1) + Q - (p(time-1)^2/(R + p(time-1)));

   % Calculate the cumulative time difference error (cum_TDE) by
   % adding the current TDE (TDE_data) to the previous value for
   % the cumulative TDE.  The current TDE is scaled by a factor of
   % minutes/60 to account for the number of corrections per hour.
   cum_TDE(time) = cum_TDE(time-1) + (TDE_data(time) * (minutes/60));

   % Output of the Kalman estimator prior to quanitization.  The two
   % parameters that are fed back are the estimate of the current TDE
   % (TDE_est) and the actual cumulative TDE (cum_TDE)
   out(time) = -L * [TDE_est(time); cum_TDE(time)];

   % LPA Decision
   %
   % The output of the Kalman estimator is quantized to interface
   % with the Loran-C transmitter timing controller which accepts
   % corrections in 20 ns intervals.  It is a mid-tread quantizer
   % with equal sized steps of 20 ns.  The step size may be varied
   % by adjusting the quantization variable, quant_var.  The output
   % of the quantizer is the linear phase adjustment (LPA).
   if out(time) >= 17*quant_var
      LPA(time) = 180;
   elseif out(time) >= 15*quant_var
      LPA(time) = 160;
   elseif out(time) >= 13*quant_var
      LPA(time) = 140;
   elseif out(time) >= 11*quant_var
      LPA(time) = 120;
```

```
   elseif out(time) >= 9*quant_var
      LPA(time) = 100;
   elseif out(time) >= 7*quant_var
      LPA(time) = 80;
   elseif out(time) >= 5*quant_var
      LPA(time) = 60;
   elseif out(time) >= 3*quant_var
      LPA(time) = 40;
   elseif out(time) >= quant_var
      LPA(time) = 20;
   elseif out(time) >= -quant_var
      LPA(time) = 0;
   elseif out(time) >= -3*quant_var
      LPA(time) = -20;
   elseif out(time) >= -5*quant_var
      LPA(time) = -40;
   elseif out(time) >= -7*quant_var
      LPA(time) = -60;
   elseif out(time) >= -9*quant_var
      LPA(time) = -80;
   elseif out(time) >= -11*quant_var
      LPA(time) = -100;
   elseif out(time) >= -13*quant_var
      LPA(time) = -120;
   elseif out(time) >= -15*quant_var
      LPA(time) = -140;
   elseif out(time) >= -17*quant_var
      LPA(time) = -160;
   elseif out(time) >= -19*quant_var
      LPA(time) = -180;
   end

end

% Generate vector to plot LPA.  Dividing by zero generates a
% vector of NaN so that only the LPA locations determined in
% the for loop below will be plotted.
plotlpa = zeros(size(LPA))/0.0;

% This for loop determines the locations of the LPAs by indexing
% the capability of Matlab to index a vector based on its non-zero
% values.  The location of each LPA is transferred to the NaN
```

```
% vector generated above.  Since Matlab does not plot NaN, the only
% values that are plotted are the LPA locations.
for index=1:data_cols
   if LPA(index)~=0
      plotlpa(index)=LPA(index);
   end
end

% Generate the LORAN Strip Charts
%
% The plots are generated using subplot so that they appear with
% similar resolution to the existing strip charts.  The plots
% include the TDE averages (TDE_data), the cumulative TDE, and the
% locations of the LPAs input by CALOC.  The factor of 0.125 on
% the cumulative error is due to the frequency of the averages
% (i.e., 7.5 minutes is 1/8 of an hour).  The LPAs are plotted
% with a multiplicative factor of 3 solely for ease of display.
xaxis = [1:num_rows/time_int]/(60/minutes);

figure
subplot(311), plot(xaxis, TDE_data, 'w+', xaxis, cum_TDE, 'w-',
                   xaxis, plotlpa*2, 'wo')
xlabel('Time (hours)'), ylabel('TDE (ns)')
title(graph_title)
axis([0 24 -80 80]), grid on
```

b. Drivers

```
========================================================================
========================================================================

data18seuscorr

graph_title = parse_title('seus_18jnY');
kalman(seus18jnY, -31, graph_title);

graph_title = parse_title('seus_18jnZ');
kalman(seus18jnZ, -77, graph_title);
```

```
data20seuscorr

graph_title = parse_title('seus_20jnW');
kalman(seus20jnW, -40, graph_title);

graph_title = parse_title('seus_20jnX');
kalman(seus20jnX, -60, graph_title);

graph_title = parse_title('seus_20jnY');
kalman(seus20jnY, -50, graph_title);

graph_title = parse_title('seus_20jnZ');
kalman(seus20jnZ, -92, graph_title);

.

data18socuscorr

graph_title = parse_title('socus18jnV');
kalman(socus18jV, -55, graph_title);

graph_title = parse_title('socus18jnW');
kalman(socus18jW, -17, graph_title);

graph_title = parse_title('socus18jnX');
kalman(socus18jX, 21, graph_title);

graph_title = parse_title('socus18jnY');
kalman(socus18jY, 16, graph_title);

graph_title = parse_title('socus18jnZ');
kalman(socus18jZ, -20, graph_title);



data20socuscorr

graph_title = parse_title('socus20jnV');
kalman(socus20jV, -42, graph_title);

graph_title = parse_title('socus20jnW');
kalman(socus20jW, -20, graph_title);
```

```
graph_title = parse_title('socus20jnX');
kalman(socus20jX, 20, graph_title);

graph_title = parse_title('socus20jnY');
kalman(socus20jY, -42, graph_title);

graph_title = parse_title('socus20jnZ');
kalman(socus20jZ, 4, graph_title);



data4nocuscorr

graph_title = parse_title('nocus04myW');
kalman(nocus4myW, -35, graph_title);

graph_title = parse_title('nocus04myX');
kalman(nocus4myX, 30, graph_title);

graph_title = parse_title('nocus04myY');
kalman(nocus4myY, 25, graph_title);



data5nocuscorr

graph_title = parse_title('nocus05myW');
kalman(nocus5myW, -15, graph_title);

graph_title = parse_title('nocus05myX');
kalman(nocus5myX, -25, graph_title);

graph_title = parse_title('nocus05myY');
kalman(nocus5myY, 5, graph_title);



data4uswccorr

graph_title = parse_title('uswc_04myW');
```

```
kalman(uswc4myW, 80, graph_title);

graph_title = parse_title('uswc_04myX');
kalman(uswc4myX, -30, graph_title);

graph_title = parse_title('uswc_04myY');
kalman(uswc4myY, -40, graph_title);



data5uswccorr

graph_title = parse_title('uswc_05myW');
kalman(uswc5myW, 60, graph_title);

graph_title = parse_title('uswc_05myX');
kalman(uswc5myX, 25, graph_title);

graph_title = parse_title('uswc_05myY');
kalman(uswc5myY, 25, graph_title);



datakodiakcorr
graph_title = parse_title('kodik11spZ');
kalman(kodiakZ, -15, graph_title);
```

5. DATA CORRECTION

```
%========================================================================
%========================================================================

% LORAN-C Thesis
%
% data18seuscorr.m
%
% Data Set:  Malone-SEUS 18Jan97
% Program removes influence of CALOC from data.  The average time
% delay was determined by observing the absolute time delay of
% each CALOC-generated LPA and taking the mean.  2 sigma outliers
% were removed to obtain the final solution of 56 time steps, or
% 560 seconds (9+ minutes).  Each LPA was removed by applying the
% opposite sign and linearly time phasing in the correction to
% obtain uncorrected data.

% Load data set
load seus18jn.dat;

% Remove bias from data.  Original data has a range of 0-99 which
% equates to the actual range of -50 to +49.  Each single step
% increase in the data equates to an actual 10 ns increase in the
% TDE, therefore the data is scaled by 10.
seus18jn = (seus18jn - 50) * 10;

% Load LORAN chain data vectors for Yankee and Zulu.
seus18jnY = seus18jn(:,8);
seus18jnZ = seus18jn(:,7);

% Determine the size of the data file.
[num_rows, num_cols] = size(seus18jn);

% Correction delay as determined by hand calculation of mean and
% variance of average LPA time delays.  The remainder of this
% section generates the vector of length 56 for a given LPA to
% linearly enter the correction.
delay = 56;
corr20 = (20/delay).*[1:delay]';

% Remove LPAs to get raw data.  The locations of the corrections
```

```
% were determined by manually determining the beginning and
% ending points of the individual LPA.  The time-phased correction
% was centered on the range of the actual delay for each LPA.
seus18jnY(667:722) = seus18jnY(667:722) + corr20;
seus18jnY(723:1120) = seus18jnY(723:1120) + 20;
seus18jnY(1121:1176) = seus18jnY(1121:1176) + 20 + corr20;
seus18jnY(1177:2133) = seus18jnY(1177:2133) + 40;
seus18jnY(2134:2189) = seus18jnY(2134:2189) + 40 + corr20;
seus18jnY(2190:3073) = seus18jnY(2190:3073) + 60;
seus18jnY(3074:3129) = seus18jnY(3074:3129) + 60 + corr20;
seus18jnY(3130:4018) = seus18jnY(3130:4018) + 80;
seus18jnY(4019:4074) = seus18jnY(4019:4074) + 80 + corr20;
seus18jnY(4075:5373) = seus18jnY(4075:5373) + 100;
seus18jnY(5374:5429) = seus18jnY(5374:5429) + 100 - corr20;
seus18jnY(5430:5902) = seus18jnY(5430:5902) + 80;
seus18jnY(5903:5958) = seus18jnY(5903:5958) + 80 - corr20;
seus18jnY(5959:6423) = seus18jnY(5959:6423) + 60;
seus18jnY(6424:6479) = seus18jnY(6424:6479) + 60 - corr20;
seus18jnY(6480:7217) = seus18jnY(6480:7217) + 40;
seus18jnY(7218:7273) = seus18jnY(7218:7273) + 40 - corr20;
seus18jnY(7274:num_rows) = seus18jnY(7274:num_rows) + 20;

seus18jnZ(2229:2284) = seus18jnZ(2229:2284) + corr20;
seus18jnZ(2285:4004) = seus18jnZ(2285:4004) + 20;
seus18jnZ(4005:4060) = seus18jnZ(4005:4060) + 20 + corr20;
seus18jnZ(4061:5370) = seus18jnZ(4061:5370) + 40;
seus18jnZ(5371:5426) = seus18jnZ(5371:5426) + 40 - corr20;
seus18jnZ(5427:5904) = seus18jnZ(5427:5904) + 20;
seus18jnZ(5905:5960) = seus18jnZ(5905:5960) + 20 - corr20;
seus18jnZ(6345:6400) = seus18jnZ(6345:6400) - corr20;
seus18jnZ(6401:6689) = seus18jnZ(6401:6689) - 20;
seus18jnZ(6690:6745) = seus18jnZ(6690:6745) - 20 - corr20;
seus18jnZ(6746:7424) = seus18jnZ(6746:7424) - 40;
seus18jnZ(7425:7480) = seus18jnZ(7425:7480) - 40 - corr20;
seus18jnZ(7481:num_rows) = seus18jnZ(7481:num_rows) - 60;



================================================================================
================================================================================

% LORAN-C Thesis
```

```
%
% data20seuscorr.m
%
% Data Set:  Malone-SEUS 20Jan97
% Program removes influence of CALOC from data.  The average time
% delay was determined by observing the absolute time delay of
% each CALOC-generated LPA and taking the mean.  2 sigma outliers
% were removed to obtain the final solution of 56 time steps, or
% 560 seconds (9+ minutes).  Each LPA was removed by applying the
% opposite sign and linearly time phasing in the correction to
% obtain uncorrected data.

% Load data set
load seus20jn.dat;

% Remove bias from data.  Original data has a range of 0-99 which
% equates to the actual range of -50 to +49.  Each single step
% increase in the data equates to an actual 10 ns increase in the
% TDE, therefore the data is scaled by 10.
seus20jn = (seus20jn - 50) * 10;

% Load LORAN chain data vectors for Whiskey, Xray, Yankee, and Zulu.
seus20jnW = seus20jn(:,1);
seus20jnX = seus20jn(:,2);
seus20jnY = seus20jn(:,8);
seus20jnZ = seus20jn(:,7);

% Determine the size of the data file.
[num_rows, num_cols] = size(seus20jn);

% Correction delay as determined by hand calculation of mean and
% variance of average LPA time delays.  The remainder of this
% section generates the vector of length 56 for a given LPA to
% linearly enter the correction.
delay = 56;
corr20 = (20/delay).*[1:delay]';
corr40 = (40/delay).*[1:delay]';

% Remove LPAs to get raw data.  The locations of the corrections
% were determined by manually determining the beginning and
% ending points of the individual LPA.  The time-phased correction
% was centered on the range of the actual delay for each LPA.
```

```
seus20jnW(1294:1349) = seus20jnW(1294:1349) + corr20;
seus20jnW(1350:1952) = seus20jnW(1350:1952) + 20;
seus20jnW(1953:2008) = seus20jnW(1953:2008) - corr20;
seus20jnW(3961:4016) = seus20jnW(3961:4016) - corr20;
seus20jnW(4017:4631) = seus20jnW(4017:4631) - 20;
seus20jnW(4632:4687) = seus20jnW(4632:4687) - 20 - corr20;
seus20jnW(4688:5511) = seus20jnW(4688:5511) - 40;
seus20jnW(5512:5567) = seus20jnW(5512:5567) - 40 - corr20;
seus20jnW(5568:7172) = seus20jnW(5568:7172) - 60;
seus20jnW(7173:7228) = seus20jnW(7173:7228) - 60 - corr20;
seus20jnW(7229:num_rows) = seus20jnW(7229:num_rows) - 80;

seus20jnX(980:1035) = seus20jnX(980:1035) + corr20;
seus20jnX(1036:1793) = seus20jnX(1036:1793) + 20;
seus20jnX(1794:1849) = seus20jnX(1794:1849) - corr20;
seus20jnX(2311:2366) = seus20jnX(2311:2366) - corr20;
seus20jnX(2367:3052) = seus20jnX(2367:3052) - 20;
seus20jnX(3053:3108) = seus20jnX(3053:3108) + corr20;
seus20jnX(4129:4184) = seus20jnX(4129:4184) + corr20;
seus20jnX(4185:4558) = seus20jnX(4185:4558) + 20;
seus20jnX(4559:4614) = seus20jnX(4559:4614) - corr20;
seus20jnX(4791:4846) = seus20jnX(4791:4846) - corr20;
seus20jnX(4847:5440) = seus20jnX(4847:5440) - 20;
seus20jnX(5441:5496) = seus20jnX(5441:5496) - 20 - corr20;
seus20jnX(5497:5608) = seus20jnX(5497:5608) - 40;
seus20jnX(5609:5664) = seus20jnX(5609:5664) - 40 - corr20;
seus20jnX(5665:6348) = seus20jnX(5665:6348) - 60;
seus20jnX(6349:6404) = seus20jnX(6349:6404) - 60 - corr20;
seus20jnX(6405:7345) = seus20jnX(6405:7345) - 80;
seus20jnX(7346:7401) = seus20jnX(7346:7401) - 80 - corr20;
seus20jnX(7402:num_rows) = seus20jnX(7402:num_rows) - 100;

seus20jnY(301:356) = seus20jnY(301:356) + corr20;
seus20jnY(357:886) = seus20jnY(357:886) + 20;
seus20jnY(887:942) = seus20jnY(887:942) + 20 + corr20;
seus20jnY(943:1547) = seus20jnY(943:1547) + 40;
seus20jnY(1548:1603) = seus20jnY(1548:1603) + 40 + corr20;
seus20jnY(1604:3126) = seus20jnY(1604:3126) + 60;
seus20jnY(3127:3182) = seus20jnY(3127:3182) + 60 + corr20;
seus20jnY(3183:3880) = seus20jnY(3183:3880) + 80;
seus20jnY(3881:3936) = seus20jnY(3881:3936) + 80 + corr20;
seus20jnY(3937:4305) = seus20jnY(3937:4305) + 100;
```

```
seus20jnY(4306:4361) = seus20jnY(4306:4361) + 100 - corr20;
seus20jnY(4362:4472) = seus20jnY(4362:4472) + 80;
seus20jnY(4473:4528) = seus20jnY(4473:4528) + 80 - corr20;
seus20jnY(4529:4800) = seus20jnY(4529:4800) + 60;
seus20jnY(4801:4856) = seus20jnY(4801:4856) + 60 - corr20;
seus20jnY(4857:4963) = seus20jnY(4857:4963) + 40;
seus20jnY(4964:5019) = seus20jnY(4964:5019) + 40 - corr20;
seus20jnY(5020:5444) = seus20jnY(5020:5444) + 20;
seus20jnY(5445:5500) = seus20jnY(5445:5500) + 20 - corr20;
seus20jnY(5532:5587) = seus20jnY(5532:5587) - corr20;
seus20jnY(5588:6357) = seus20jnY(5588:6357) - 20;
seus20jnY(6358:6413) = seus20jnY(6358:6413) - 20 - corr20;
seus20jnY(6414:7401) = seus20jnY(6414:7401) - 40;
seus20jnY(7402:7457) = seus20jnY(7402:7457) - 40 - corr20;
seus20jnY(7458:num_rows) = seus20jnY(7458:num_rows) - 60;

seus20jnZ(1229:1284) = seus20jnZ(1229:1284) + corr20;
seus20jnZ(1285:2167) = seus20jnZ(1285:2167) + 20;
seus20jnZ(2168:2223) = seus20jnZ(2168:2223) + 20 + corr20;
seus20jnZ(2224:3881) = seus20jnZ(2224:3881) + 40;
seus20jnZ(3882:3937) = seus20jnZ(3882:3937) + 40 - corr20;
seus20jnZ(3938:4399) = seus20jnZ(3938:4399) + 20;
seus20jnZ(4400:4455) = seus20jnZ(4400:4455) + 20 - corr20;
seus20jnZ(4647:4702) = seus20jnZ(4647:4702) - corr20;
seus20jnZ(4703:4897) = seus20jnZ(4703:4897) - 20;
seus20jnZ(4898:4953) = seus20jnZ(4898:4953) - 20 - corr20;
seus20jnZ(4954:5460) = seus20jnZ(4954:5460) - 40;
seus20jnZ(5461:5516) = seus20jnZ(5461:5516) - 40 - corr20;
seus20jnZ(5517:5625) = seus20jnZ(5517:5625) - 60;
seus20jnZ(5626:5681) = seus20jnZ(5626:5681) - 60 - corr20;
seus20jnZ(5682:5712) = seus20jnZ(5682:5712) - 80;
seus20jnZ(5713:5768) = seus20jnZ(5713:5768) - 80 - corr20;
seus20jnZ(5769:5886) = seus20jnZ(5769:5886) - 100;
seus20jnZ(5887:5942) = seus20jnZ(5887:5942) - 100 - corr20;
seus20jnZ(5943:6267) = seus20jnZ(5943:6267) - 120;
seus20jnZ(6268:6323) = seus20jnZ(6268:6323) - 120 - corr20;
seus20jnZ(6324:7332) = seus20jnZ(6324:7332) - 140;
seus20jnZ(7333:7388) = seus20jnZ(7333:7388) - 140 - corr20;
seus20jnZ(7389:num_rows) = seus20jnZ(7389:num_rows) - 160;
```

```
================================================================
================================================================

% LORAN-C Thesis
%
% data18socuscorr.m
%
% Data Set:  Malone-SOCUS 18Jan97
% Program removes influence of CALOC from data.  The average time
% delay was determined by observing the absolute time delay of
% each CALOC-generated LPA and taking the mean.  2 sigma outliers
% were removed to obtain the final solution of 56 time steps, or
% 560 seconds (9+ minutes).  Each LPA was removed by applying the
% opposite sign and linearly time phasing in the correction to
% obtain uncorrected data.

% Load data set
load socus18j.dat;

% Remove bias from data.  Original data has a range of 0-99 which
% equates to the actual range of -50 to +49.  Each single step
% increase in the data equates to an actual 10 ns increase in the
% TDE, therefore the data is scaled by 10.
socus18j = (socus18j - 50) * 10;

% Load LORAN chain data vectors for Victor, Whiskey, Xray, Yankee,
% and Zulu.
socus18jV = socus18j(:,1);
socus18jW = socus18j(:,2);
socus18jX = socus18j(:,4);
socus18jY = socus18j(:,6);
socus18jZ = socus18j(:,9);

% Determine the size of the data file.
[num_rows, num_cols] = size(socus18j);

% Correction delay as determined by hand calculation of mean and
% variance of average LPA time delays.  The remainder of this
% section generates the vector of length 56 for a given LPA to
% linearly enter the correction.
delay = 56;
corr20 = (20/delay).*[1:delay]';
```

```
corr40 = (40/delay).*[1:delay]';

% Remove LPAs to get raw data.  The locations of the corrections
% were determined by manually determining the beginning and
% ending points of the individual LPA.  The time-phased correction
% was centered on the range of the actual delay for each LPA.
socus18jV(2667:2722) = socus18jV(2667:2722) - corr20;
socus18jV(2723:4179) = socus18jV(2723:4179) - 20;
socus18jV(4180:4235) = socus18jV(4180:4235) + corr20;
socus18jV(5923:5978) = socus18jV(5923:5978) - corr20;
socus18jV(5979:6883) = socus18jV(5979:6883) - 20;
socus18jV(6884:6939) = socus18jV(6884:6939) - 20 - corr20;
socus18jV(6940:7799) = socus18jV(6940:7799) - 40;
socus18jV(7800:7855) = socus18jV(7800:7855) - 40 - corr20;
socus18jV(7856:num_rows) = socus18jV(7856:num_rows) - 60;

socus18jW(1736:1791) = socus18jW(1736:1791) + corr20;
socus18jW(1792:2850) = socus18jW(1792:2850) + 20;
socus18jW(2851:2906) = socus18jW(2851:2906) - corr20;
socus18jW(5313:5368) = socus18jW(5313:5368) - corr20;
socus18jW(5369:5999) = socus18jW(5369:5999) - 20;
socus18jW(6000:6055) = socus18jW(6000:6055) - 20 - corr20;
socus18jW(6056:6798) = socus18jW(6056:6798) - 40;
socus18jW(6799:6854) = socus18jW(6799:6854) - 40 - corr20;
socus18jW(6855:num_rows) = socus18jW(6855:num_rows) - 60;

socus18jX(1046:1101) = socus18jX(1046:1101) - corr20;
socus18jX(1102:num_rows) = socus18jX(1102:num_rows) - 20;

socus18jY(1121:1176) = socus18jY(1121:1176) - corr20;
socus18jY(1177:1317) = socus18jY(1177:1317) - 20;
socus18jY(1318:1373) = socus18jY(1318:1373) - 20 - corr20;
socus18jY(1374:1727) = socus18jY(1374:1727) - 40;
socus18jY(1728:1783) = socus18jY(1728:1783) - 40 + corr20;
socus18jY(1784:5929) = socus18jY(1784:5929) - 20;
socus18jY(5930:5985) = socus18jY(5930:5985) - 20 + corr20;

socus18jZ(543:598) = socus18jZ(543:598) - corr20;
socus18jZ(599:1821) = socus18jZ(599:1821) - 20;
socus18jZ(1825:1880) = socus18jZ(1825:1880) - 20 - corr20;
socus18jZ(1881:2903) = socus18jZ(1881:2903) - 40;
socus18jZ(2904:2959) = socus18jZ(2904:2959) - 40 - corr20;
```

```
socus18jZ(2960:4298) = socus18jZ(2960:4298) - 60;
socus18jZ(4299:4354) = socus18jZ(4299:4354) - 60 - corr20;
socus18jZ(4355:4985) = socus18jZ(4355:4985) - 80;
socus18jZ(4986:5041) = socus18jZ(4986:5041) - 80 + corr20;
socus18jZ(5042:5076) = socus18jZ(5042:5076) - 60;
socus18jZ(5077:5132) = socus18jZ(5077:5132) - 60 + corr20;
socus18jZ(5133:5806) = socus18jZ(5133:5806) - 40;
socus18jZ(5807:5862) = socus18jZ(5807:5862) - 40 + corr40;
socus18jZ(6418:6473) = socus18jZ(6418:6473) - corr20;
socus18jZ(6474:7311) = socus18jZ(6474:7311) - 20;
socus18jZ(7312:7367) = socus18jZ(7312:7367) - 20 - corr20;
socus18jZ(7368:num_rows) = socus18jZ(7368:num_rows) - 40;
```

==========

==========

```
% LORAN-C Thesis
%
% data20socuscorr.m
%
% Data Set:  Malone-SOCUS 20Jan97
% Program removes influence of CALOC from data.  The average time
% delay was determined by observing the absolute time delay of
% each CALOC-generated LPA and taking the mean.  2 sigma outliers
% were removed to obtain the final solution of 56 time steps, or
% 560 seconds (9+ minutes).  Each LPA was removed by applying the
% opposite sign and linearly time phasing in the correction to
% obtain uncorrected data.

% Load data set
load socus20j.dat;

% Remove bias from data.  Original data has a range of 0-99 which
% equates to the actual range of -50 to +49.  Each single step
% increase in the data equates to an actual 10 ns increase in the
% TDE, therefore the data is scaled by 10.
socus20j = (socus20j - 50) * 10;

% Load LORAN chain data vectors for Victor, Whiskey, Xray, Yankee,
% and Zulu.
```

```
socus20jV = socus20j(:,1);
socus20jW = socus20j(:,2);
socus20jX = socus20j(:,4);
socus20jY = socus20j(:,6);
socus20jZ = socus20j(:,9);

% Determine the size of the data file.
[num_rows, num_cols] = size(socus20j);

% Correction delay as determined by hand calculation of mean and
% variance of average LPA time delays.  The remainder of this
% section generates the vector of length 56 for a given LPA to
% linearly enter the correction.
delay = 56;
corr20 = (20/delay).*[1:delay]';
corr40 = (40/delay).*[1:delay]';

% Remove LPAs to get raw data.  The locations of the corrections
% were determined by manually determining the beginning and
% ending points of the individual LPA.  The time-phased correction
% was centered on the range of the actual delay for each LPA.
socus20jV(222:277) = socus20jV(222:277) - corr20;
socus20jV(278:6029) = socus20jV(278:6029) - 20;
socus20jV(6030:6085) = socus20jV(6030:6085) - 20 - corr20;
socus20jV(6086:num_rows) = socus20jV(6086:num_rows) - 40;

socus20jW(368:423) = socus20jW(368:423) - corr20;
socus20jW(424:617) = socus20jW(424:617) - 20;
socus20jW(618:673) = socus20jW(618:673) - 20 - corr20;
socus20jW(674:1375) = socus20jW(674:1375) - 40;
socus20jW(1376:1431) = socus20jW(1376:1431) - 40 + corr20;
socus20jW(1432:3061) = socus20jW(1432:3061) - 20;
socus20jW(3062:3117) = socus20jW(3062:3117) - 20 + corr20;
socus20jW(3916:3971) = socus20jW(3916:3971) - corr20;
socus20jW(3972:6423) = socus20jW(3972:6423) - 20;
socus20jW(6424:6479) = socus20jW(6424:6479) - 20 - corr20;
socus20jW(6480:num_rows) = socus20jW(6480:num_rows) - 40;

socus20jX(2701:2756) = socus20jX(2701:2756) - corr20;
socus20jX(2757:num_rows) = socus20jX(2757:num_rows) - 20;

socus20jY(2870:2925) = socus20jY(2870:2925) - corr20;
```

```
socus20jY(2926:5627) = socus20jY(2926:5627) - 20;
socus20jY(5628:5683) = socus20jY(5628:5683) - 20 - corr20;
socus20jY(5684:6762) = socus20jY(5684:6762) - 40;
socus20jY(6763:6818) = socus20jY(6763:6818) - 40 + corr20;
socus20jY(6819:num_rows) = socus20jY(6819:num_rows) - 20;

socus20jZ(349:404) = socus20jZ(349:404) - corr20;
socus20jZ(405:3458) = socus20jZ(405:3458) - 20;
socus20jZ(3459:3514) = socus20jZ(3459:3514) - 20 - corr20;
socus20jZ(3515:4083) = socus20jZ(3515:4083) - 40;
socus20jZ(4084:4139) = socus20jZ(4084:4139) - 40 - corr20;
socus20jZ(4140:4691) = socus20jZ(4140:4691) - 60;
socus20jZ(4692:4747) = socus20jZ(4692:4747) - 60 + corr20;
socus20jZ(4748:6450) = socus20jZ(4748:6450) - 40;
socus20jZ(6451:6506) = socus20jZ(6451:6506) - 40 + corr20;
socus20jZ(6507:7089) = socus20jZ(6507:7089) - 20;
socus20jZ(7090:7145) = socus20jZ(7090:7145) - 20 + corr20;
```

```
================================================================================
================================================================================

% LORAN-C Thesis
%
% data4nocuscorr.m
%
% Data Set:  Middletown-NOCUS 4May97
% Program removes influence of CALOC from data.  The average time
% delay was determined by observing the absolute time delay of
% each CALOC-generated LPA and taking the mean.  2 sigma outliers
% were removed to obtain the final solution of 56 time steps, or
% 560 seconds (9+ minutes).  Each LPA was removed by applying the
% opposite sign and linearly time phasing in the correction to
% obtain uncorrected data.

% Load data set
load nocus4my.dat;

% Remove bias from data.  Original data has a range of 0-99 which
% equates to the actual range of -50 to +49.  Each single step
% increase in the data equates to an actual 10 ns increase in the
```

```
% TDE, therefore the data is scaled by 10.
nocus4my = (nocus4my - 50) * 10;

% Load LORAN chain data vectors for Whiskey, Xray, Yankee.
nocus4myW = nocus4my(:,10);
nocus4myX = nocus4my(:,12);
nocus4myY = nocus4my(:,3);

% Determine the size of the data file.
[num_rows, num_cols] = size(nocus4my);

% Correction delay as determined by hand calculation of mean and
% variance of average LPA time delays.  The remainder of this
% section generates the vector of length 56 for a given LPA to
% linearly enter the correction.
delay = 56;
corr20 = (20/delay).*[1:delay]';
corr40 = (40/delay).*[1:delay]';

% Remove LPAs to get raw data.  The locations of the corrections
% were determined by manually determining the beginning and
% ending points of the individual LPA.  The time-phased correction
% was centered on the range of the actual delay for each LPA.
nocus4myW(565:620) = nocus4myW(565:620) - corr40;
nocus4myW(621:1033) = nocus4myW(621:1033) - 40;
nocus4myW(1034:1089) = nocus4myW(1034:1089) - 40 - corr20;
nocus4myW(1090:3507) = nocus4myW(1090:3507) - 60;
nocus4myW(3508:3563) = nocus4myW(3508:3563) - 60 - corr20;
nocus4myW(3564:4844) = nocus4myW(3564:4844) - 80;
nocus4myW(4845:4900) = nocus4myW(4845:4900) - 80 + corr20;
nocus4myW(4901:5922) = nocus4myW(4901:5922) - 60;
nocus4myW(5923:5978) = nocus4myW(5923:5978) - 60 + corr20;
nocus4myW(5979:7039) = nocus4myW(5979:7039) - 40;
nocus4myW(7040:7095) = nocus4myW(7040:7095) - 40 + corr20;
nocus4myW(7096:7951) = nocus4myW(7096:7951) - 20;
nocus4myW(7952:8007) = nocus4myW(7952:8007) - 20 - corr20;
nocus4myW(8008:num_rows) = nocus4myW(8008:num_rows) - 40;

nocus4myX(2136:2191) = nocus4myX(2136:2191) - corr20;
nocus4myX(2192:6456) = nocus4myX(2192:6456) - 20;
nocus4myX(6457:6512) = nocus4myX(6457:6512) - 20 + corr20;
nocus4myX(7047:7102) = nocus4myX(7047:7102) + corr20;
```

```
nocus4myX(7103:7976) = nocus4myX(7103:7976) + 20;
nocus4myX(7977:8032) = nocus4myX(7977:8032) + 20 - corr20;

nocus4myY(1715:1770) = nocus4myY(1715:1770) - corr20;
nocus4myY(1771:2213) = nocus4myY(1771:2213) - 20;
nocus4myY(2214:2269) = nocus4myY(2214:2269) - 20 - corr20;
nocus4myY(2270:2650) = nocus4myY(2270:2650) - 40;
nocus4myY(2651:2706) = nocus4myY(2651:2706) - 40 - corr40;
nocus4myY(2707:5650) = nocus4myY(2707:5650) - 80;
nocus4myY(5651:5706) = nocus4myY(5651:5706) - 80 + corr20;
nocus4myY(5707:6525) = nocus4myY(5707:6525) - 60;
nocus4myY(6526:6581) = nocus4myY(6526:6581) - 60 + corr20;
nocus4myY(6582:7035) = nocus4myY(6582:7035) - 40;
nocus4myY(7036:7091) = nocus4myY(7036:7091) - 40 + corr20;
nocus4myY(7092:7966) = nocus4myY(7092:7966) - 20;
nocus4myY(7967:8022) = nocus4myY(7967:8022) - 20 + corr20;
```

```
================================================================
================================================================

% LORAN-C Thesis
%
% data5nocuscorr.m
%
% Data Set:  Middletown-NOCUS 5May97
% Program removes influence of CALOC from data.  The average time
% delay was determined by observing the absolute time delay of
% each CALOC-generated LPA and taking the mean.  2 sigma outliers
% were removed to obtain the final solution of 56 time steps, or
% 560 seconds (9+ minutes).  Each LPA was removed by applying the
% opposite sign and linearly time phasing in the correction to
% obtain uncorrected data.

% Load data set
load nocus5my.dat;

% Remove bias from data.  Original data has a range of 0-99 which
% equates to the actual range of -50 to +49.  Each single step
% increase in the data equates to an actual 10 ns increase in the
% TDE, therefore the data is scaled by 10.
```

```
nocus5my = (nocus5my - 50) * 10;

% Load LORAN chain data vectors for Whiskey, Xray, Yankee.
nocus5myW = nocus5my(:,10);
nocus5myX = nocus5my(:,12);
nocus5myY = nocus5my(:,3);

% Determine the size of the data file.
[num_rows, num_cols] = size(nocus5my);

% Correction delay as determined by hand calculation of mean and
% variance of average LPA time delays.  The remainder of this
% section generates the vector of length 56 for a given LPA to
% linearly enter the correction.
delay = 56;
corr20 = (20/delay).*[1:delay]';
corr40 = (40/delay).*[1:delay]';

% Remove LPAs to get raw data.  The locations of the corrections
% were determined by manually determining the beginning and
% ending points of the individual LPA.  The time-phased correction
% was centered on the range of the actual delay for each LPA.
nocus5myW(6554:6609) = nocus5myW(6554:6609) + corr20;
nocus5myW(6610:num_rows) = nocus5myW(6610:num_rows) + 20;

nocus5myX(523:578) = nocus5myX(523:578) - corr20;
nocus5myX(579:4687) = nocus5myX(579:4687) - 20;
nocus5myX(4688:4743) = nocus5myX(4688:4743) -20 + corr20;

nocus5myY(530:585) = nocus5myY(530:585) - corr20;
nocus5myY(586:1309) = nocus5myY(586:1309) - 20;
nocus5myY(1310:1365) = nocus5myY(1310:1365) - 20 - corr20;
nocus5myY(1366:2213) = nocus5myY(1366:2213) - 40;
nocus5myY(2214:2269) = nocus5myY(2214:2269) - 40 - corr20;
nocus5myY(2270:3301) = nocus5myY(2270:3301) - 60;
nocus5myY(3302:3357) = nocus5myY(3302:3357) - 60 - corr20;
nocus5myY(3358:5124) = nocus5myY(3358:5124) - 80;
nocus5myY(5125:5180) = nocus5myY(5125:5180) - 80 + corr20;
nocus5myY(5181:6560) = nocus5myY(5181:6560) - 60;
nocus5myY(6561:6616) = nocus5myY(6561:6616) - 60 + corr40;
nocus5myY(6617:num_rows) = nocus5myY(6617:num_rows) - 20;
```

```
================================================================
================================================================

% LORAN-C Thesis
%
% data4uswccorr.m
%
% Data Set:  Middletown-USWC 4May97
% Program removes influence of CALOC from data.  The average time
% delay was determined by observing the absolute time delay of
% each CALOC-generated LPA and taking the mean.  2 sigma outliers
% were removed to obtain the final solution of 56 time steps, or
% 560 seconds (9+ minutes).  Each LPA was removed by applying the
% opposite sign and linearly time phasing in the correction to
% obtain uncorrected data.

% Load data set
load uswc4my.dat;

% Remove bias from data.  Original data has a range of 0-99 which
% equates to the actual range of -50 to +49.  Each single step
% increase in the data equates to an actual 10 ns increase in the
% TDE, therefore the data is scaled by 10.
uswc4my = (uswc4my - 50) * 10;

% Load LORAN chain data vectors for Whiskey, Xray, Yankee.
uswc4myW = uswc4my(:,1);
uswc4myX = uswc4my(:,10);
uswc4myY = uswc4my(:,12);

% Determine the size of the data file.
[num_rows, num_cols] = size(uswc4my);

% Correction delay as determined by hand calculation of mean and
% variance of average LPA time delays.  The remainder of this
% section generates the vector of length 56 for a given LPA to
% linearly enter the correction.
delay = 56;
corr20 = (20/delay).*[1:delay]';

% Remove LPAs to get raw data.  The locations of the corrections
```

```
% were determined by manually determining the beginning and
% ending points of the individual LPA.  The time-phased correction
% was centered on the range of the actual delay for each LPA.
uswc4myW(175:230) = uswc4myW(175:230) + corr20;
uswc4myW(231:2027) = uswc4myW(231:2027) + 20;
uswc4myW(2028:2083) = uswc4myW(2028:2083) + 20 + corr20;
uswc4myW(2084:2205) = uswc4myW(2084:2205) + 40;
uswc4myW(2206:2261) = uswc4myW(2206:2261) + 40 + corr20;
uswc4myW(2262:3880) = uswc4myW(2262:3880) + 60;
uswc4myW(3881:3936) = uswc4myW(3881:3936) + 60 + corr20;
uswc4myW(3937:4203) = uswc4myW(3937:4203) + 80;
uswc4myW(4204:4259) = uswc4myW(4204:4259) + 80 - corr20;
uswc4myW(4260:4661) = uswc4myW(4260:4661) + 60;
uswc4myW(4662:4717) = uswc4myW(4662:4717) + 60 - corr20;
uswc4myW(4718:num_rows) = uswc4myW(4718:num_rows) + 40;

uswc4myX(1515:1570) = uswc4myX(1515:1570) - corr20;
uswc4myX(1571:2200) = uswc4myX(1571:2200) - 20;
uswc4myX(2201:2256) = uswc4myX(2201:2256) - 20 - corr20;
uswc4myX(2257:3076) = uswc4myX(2257:3076) - 40;
uswc4myX(3077:3132) = uswc4myX(3077:3132) - 40 + corr20;
uswc4myX(3133:6897) = uswc4myX(3133:6897) - 20;
uswc4myX(6898:6953) = uswc4myX(6898:6953) - 20 - corr20;
uswc4myX(6954:num_rows) = uswc4myX(6954:num_rows) - 40;

uswc4myY(1751:1806) = uswc4myY(1751:1806) - corr20;
uswc4myY(1807:2437) = uswc4myY(1807:2437) - 20;
uswc4myY(2438:2493) = uswc4myY(2438:2493) - 20 - corr20;
uswc4myY(2494:3733) = uswc4myY(2494:3733) - 40;
uswc4myY(3734:3789) = uswc4myY(3734:3789) - 40 + corr20;
uswc4myY(3790:num_rows) = uswc4myY(3790:num_rows) - 20;



=========================================================================
=========================================================================

% LORAN-C Thesis
%
% data5uswccorr.m
%
% Data Set:  Middletown-USWC 5May97
```

```
% Program removes influence of CALOC from data.  The average time
% delay was determined by observing the absolute time delay of
% each CALOC-generated LPA and taking the mean.  2 sigma outliers
% were removed to obtain the final solution of 56 time steps, or
% 560 seconds (9+ minutes).  Each LPA was removed by applying the
% opposite sign and linearly time phasing in the correction to
% obtain uncorrected data.

% Load data set
load uswc5my.dat;

% Remove bias from data.  Original data has a range of 0-99 which
% equates to the actual range of -50 to +49.  Each single step
% increase in the data equates to an actual 10 ns increase in the
% TDE, therefore the data is scaled by 10.
uswc5my = (uswc5my - 50) * 10;

% Load LORAN chain data vectors for Whiskey, Xray, Yankee.
uswc5myW = uswc5my(:,1);
uswc5myX = uswc5my(:,10);
uswc5myY = uswc5my(:,12);

% Determine the size of the data file.
[num_rows, num_cols] = size(uswc5my);

% Correction delay as determined by hand calculation of mean and
% variance of average LPA time delays.  The remainder of this
% section generates the vector of length 56 for a given LPA to
% linearly enter the correction.
delay = 56;
corr20 = (20/delay).*[1:delay]';
corr40 = (40/delay).*[1:delay]';

% Remove LPAs to get raw data.  The locations of the corrections
% were determined by manually determining the beginning and
% ending points of the individual LPA.  The time-phased correction
% was centered on the range of the actual delay for each LPA.
uswc5myW(2016:2071) = uswc5myW(2016:2071) + corr20;
uswc5myW(2072:2726) = uswc5myW(2072:2726) + 20;
uswc5myW(2727:2782) = uswc5myW(2727:2782) + 20 + corr20;
uswc5myW(2783:3366) = uswc5myW(2783:3366) + 40;
uswc5myW(3367:3422) = uswc5myW(3367:3422) + 40 + corr20;
```

```
uswc5myW(3423:4202) = uswc5myW(3423:4202) + 60;
uswc5myW(4203:4258) = uswc5myW(4203:4258) + 60 - corr20;
uswc5myW(4259:5529) = uswc5myW(4259:5529) +40;
uswc5myW(5530:5585) = uswc5myW(5530:5585) +40 - corr20;
uswc5myW(5586:num_rows) = uswc5myW(5586:num_rows) + 20;

uswc5myX(1470:1525) = uswc5myX(1470:1525) - corr20;
uswc5myX(1526:3341) = uswc5myX(1526:3341) - 20;
uswc5myX(3342:3397) = uswc5myX(3342:3397) - 20 + corr40;
uswc5myX(3398:3610) = uswc5myX(3398:3610) + 20;
uswc5myX(3611:3666) = uswc5myX(3611:3666) + 20 - corr20;
uswc5myX(4197:4252) = uswc5myX(4197:4252) - corr20;
uswc5myX(4253:num_rows) = uswc5myX(4253:num_rows) - 20;

uswc5myY(2122:2177) = uswc5myY(2122:2177) - corr20;
uswc5myY(2178:3345) = uswc5myY(2178:3345) - 20;
uswc5myY(3346:3401) = uswc5myY(3346:3401) - 20 + corr40;
uswc5myY(3402:4174) = uswc5myY(3402:4174) + 20;
uswc5myY(4175:4230) = uswc5myY(4175:4230) + 20 - corr20;
uswc5myY(7021:7076) = uswc5myY(7021:7076) - corr20;
uswc5myY(7077:num_rows) = uswc5myY(7077:num_rows) - 20;



================================================================
================================================================

% LORAN-C Thesis
%
% datakodiakcorr.m
%
% Data Set:  Kodiak Zulu
% Program removes influence of CALOC from data.  The average time
% delay was determined by observing the absolute time delay of
% each CALOC-generated LPA and taking the mean.  2 sigma outliers
% were removed to obtain the final solution of 56 time steps, or
% 560 seconds (9+ minutes).  Each LPA was removed by applying the
% opposite sign and linearly time phasing in the correction to
% obtain uncorrected data.

% Load data set
load kodiak.dat;
```

```
% Remove bias from data.  Original data has a range of 0-99 which
% equates to the actual range of -50 to +49.  Each single step
% increase in the data equates to an actual 10 ns increase in the
% TDE, therefore the data is scaled by 10.
kodiak = (kodiak - 50) * 10;

% Load LORAN chain data vector for Zulu.
kodiakZ = kodiak(:,1);

% Determine the size of the data file.
[num_rows, num_cols] = size(kodiak);

% Correction delay as determined by hand calculation of mean and
% variance of average LPA time delays.  The remainder of this
% section generates the vector of length 56 for a given LPA to
% linearly enter the correction.
delay = 56;
corr20 = (20/delay).*[1:delay]';

% Remove LPAs to get raw data.  The locations of the corrections
% were determined by manually determining the beginning and
% ending points of the individual LPA.  The time-phased correction
% was centered on the range of the actual delay for each LPA.
kodiakZ(343:398) = kodiakZ(343:398) - corr20;
kodiakZ(399:1687) = kodiakZ(399:1687) - 20;
kodiakZ(1688:1743) = kodiakZ(1688:1743) - 20 + corr20;
kodiakZ(1981:2036) = kodiakZ(1981:2036) - corr20;
kodiakZ(2037:3619) = kodiakZ(2037:3619) - 20;
kodiakZ(3620:3675) = kodiakZ(3620:3675) - 20 + corr20;
```

6. CALOC STRIP CHARTS

```
%=========================================================================
%=========================================================================

% LORAN-C Thesis
%
% seus18.m
%
% Data Set:  Malone-SEUS 18Jan97
% Program generates LORAN Time Difference Error (TDE) vs Time strip
% charts similar to the output of the current CALOC strip charts.
% These strip charts will be used to compare the output of the new
% control algorithm to determine the increase in control accuracy.

% Clear stored variables
clear

% Load data set
load seus18jn.dat;

% Remove bias from data.  Original data has a range of 0-99 which
% equates to the actual range of -50 to +49.  Each single step
% increase in the data equates to an actual 10 ns increase in the
% TDE, therefore the data is scaled by 10.
seus18jn = (seus18jn - 50) * 10;

% Load LORAN chain data vectors for Yankee and Zulu.
seus18jnY = seus18jn(:,8);
seus18jnZ = seus18jn(:,7);

% Determine the size of the data file.
[num_rows, num_cols] = size(seus18jn);

% Determine number of data points after taking 7.5 minute averages.
% Since each data point represents 10 seconds in real time, there
% are 45 data points in each 7.5 minute average.  Using the Matlab
% floor command rounds down the result of the operation to get an
% integer value to use in reshaping the data for averaging.  The
% second line generates the vector for plotting the data.
data_cols = floor(num_rows/45);
xaxis = [1:num_rows/45]/8;
```

```
% Take 7.5 minute averages to generate data points for plotting
% similar to current strip charts.  Averages are taken by reshaping
% the data into a matrix of dimensions data_cols x 45 and then taking
% the mean in a column-wise fashion.  This generates a vector of
% length data_cols of the 7.5 minute averages.
avgY = mean(reshape(seus18jnY(1:(45*data_cols)), 45, data_cols));
avgZ = mean(reshape(seus18jnZ(1:(45*data_cols)), 45, data_cols));

% After plotting the raw data, the locations of the LPAs were
% determined by comparing the generated plot with the actual strip
% charts.  The locations are placed below to generate LPA vectors.
lpay = zeros(size(1:data_cols));
lpaz = zeros(size(1:data_cols));

lpay(16) = -20; lpay(26) = -20; lpay(48) = -20;
lpay(69) = -20; lpay(90) = -20;
lpay(120) = 20; lpay(132) = 20;
lpay(144) = 20; lpay(161) = 20;

lpaz(50) = -20; lpaz(90) = -20;
lpaz(120) = 20; lpaz(132) = 20; lpaz(142) = 20;
lpaz(149) = 20; lpaz(166) = 20;

% Generate vector to plot LPA.  Dividing by zero generates a
% vector of NaN so that only the LPA locations determined in
% the for loop below will be plotted.
plotlpay = zeros(size(lpay))/0.0;
plotlpaz = zeros(size(lpaz))/0.0;

% This for loop determines the locations of the LPAs by indexing
% the capability of Matlab to index a vector based on its non-zero
% values.  The location of each LPA is transferred to the NaN
% vector generated above.  Since Matlab does not plot NaN, the only
% values that are plotted are the LPA locations.
for index=1:data_cols
   if lpay(index)~=0
      plotlpay(index)=lpay(index);
   end
   if lpaz(index)~=0
      plotlpaz(index)=lpaz(index);
   end
```

```
end

% Generate the LORAN Strip Charts
%
% The plots are generated using subplot so that they appear with
% similar resolution to the existing strip charts.  The plots
% include the TDE averages (avgY), the cumulative TDE, and the
% locations of the LPAs input by CALOC.  The initial value for
% the cumulative error is taken from the actual strip charts from
% CALOC.  The factor of 0.125 on the cumulative error is due to
% the frequency of the averages (i.e., 7.5 minutes is 1/8 of an
% hour).  The LPAs are plotted with a multiplicative factor of 3
% solely for ease of display.
figure
subplot(311), plot(xaxis, avgY, 'w+')
xlabel('Time (hours)'), ylabel('TDE (ns)')
hold on
plot(xaxis, -31 + cumsum(avgY * 0.125), 'w-')
plot(xaxis, plotlpay * 3, 'wo')
grid on
axis([0 24 -80 80])
title('SEUS Yankee 18Jan - Plot of TDE vs Time (Strip Chart)')

figure
subplot(311), plot(xaxis, avgZ, 'w+')
xlabel('Time (hours)'), ylabel('TDE (ns)')
hold on
plot(xaxis, -77 + cumsum(avgZ * 0.125), 'w-')
plot(xaxis, plotlpaz * 3, 'wo')
grid on
axis([0 24 -80 80])
title('SEUS Zulu 18Jan - Plot of TDE vs Time (Strip Chart')



================================================================================
================================================================================

% LORAN-C Thesis
%
% seus20.m
%
```

```
% Data Set:  Malone-SEUS 20Jan97
% Program generates LORAN Time Difference Error (TDE) vs Time strip
% charts similar to the output of the current CALOC strip charts.
% These strip charts will be used to compare the output of the new
% control algorithm to determine the increase in control accuracy.

% Clear stored variables
clear

% Load data set
load seus20jn.dat;

% Remove bias from data.  Original data has a range of 0-99 which
% equates to the actual range of -50 to +49.  Each single step
% increase in the data equates to an actual 10 ns increase in the
% TDE, therefore the data is scaled by 10.
seus20jn = (seus20jn - 50) * 10;

% Load LORAN chain data vectors for Whiskey, Xray, Yankee, and Zulu.
seus20jnW = seus20jn(:,1);
seus20jnX = seus20jn(:,2);
seus20jnY = seus20jn(:,8);
seus20jnZ = seus20jn(:,7);

% Determine the size of the data file.
[num_rows, num_cols] = size(seus20jn);

% Determine number of data points after taking 7.5 minute averages.
% Since each data point represents 10 seconds in real time, there
% are 45 data points in each 7.5 minute average.  Using the Matlab
% floor command rounds down the result of the operation to get an
% integer value to use in reshaping the data for averaging.  The
% second line generates the vector for plotting the data.
data_cols = floor(num_rows/45);
xaxis = [1:num_rows/45]/8;

% Take 7.5 minute averages to generate data points for plotting
% similar to current strip charts.  Averages are taken by reshaping
% the data into a matrix of dimensions data_cols x 45 and then taking
% the mean in a column-wise fashion.  This generates a vector of
% length data_cols of the 7.5 minute averages.
avgW = mean(reshape(seus20jnW(1:(45*data_cols)), 45, data_cols));
```

```
avgX = mean(reshape(seus20jnX(1:(45*data_cols)), 45, data_cols));
avgY = mean(reshape(seus20jnY(1:(45*data_cols)), 45, data_cols));
avgZ = mean(reshape(seus20jnZ(1:(45*data_cols)), 45, data_cols));

% After plotting the raw data, the locations of the LPAs were
% determined by comparing the generated plot with the actual strip
% charts.  The locations are placed below to generate LPA vectors.
lpaw = zeros(size(1:num_rows/45));
lpax = zeros(size(1:num_rows/45));
lpay = zeros(size(1:num_rows/45));
lpaz = zeros(size(1:num_rows/45));

lpaw(30) = -20;

lpaw(44) = 20; lpaw(89) = 20; lpaw(104) = 20;
lpaw(123) = 20; lpaw(160) = 20;

lpax(22) = -20; lpax(68) = -20; lpax(92) = -20;

lpax(41) = 20; lpax(52) = 20; lpax(102) = 20;
lpax(107) = 20; lpax(122) = 20; lpax(125) = 20;
lpax(142) = 20; lpax(164) = 20;

lpay(7) = -20; lpay(20) = -20; lpay(35) = -20;
lpay(70) = -20; lpay(87) = -20;

lpay(96) = 20; lpay(100) = 20; lpay(107) = 40;
lpay(111) = 20; lpay(122) = 20; lpay(124) = 40;
lpay(142) = 20; lpay(164) = 20;

lpaz(28) = -20; lpaz(49) = -20;

lpaz(87) = 20; lpaz(98) = 20; lpaz(104) = 20;
lpaz(110) = 20; lpaz(122) = 20; lpaz(126) = 20;
lpaz(128) = 20; lpaz(131) = 20; lpaz(140) = 20;
lpaz(164) = 20;

% Generate vector to plot LPA.  Dividing by zero generates a
% vector of NaN so that only the LPA locations determined in
% the for loop below will be plotted.
plotlpaw = zeros(size(lpaw))/0.0;
plotlpax = zeros(size(lpax))/0.0;
```

```
plotlpay = zeros(size(lpay))/0.0;
plotlpaz = zeros(size(lpaz))/0.0;

% This for loop determines the locations of the LPAs by indexing
% the capability of Matlab to index a vector based on its non-zero
% values.  The location of each LPA is transferred to the NaN
% vector generated above.  Since Matlab does not plot NaN, the only
% values that are plotted are the LPA locations.
for index = 1:data_cols
   if lpaw(index)~=0
      plotlpaw(index)=lpaw(index);
   end
   if lpax(index)~=0
      plotlpax(index)=lpax(index);
   end
   if lpay(index)~=0
      plotlpay(index)=lpay(index);
   end
   if lpaz(index)~=0
      plotlpaz(index)=lpaz(index);
   end
end

% Generate the LORAN Strip Charts
%
% The plots are generated using subplot so that they appear with
% similar resolution to the existing strip charts.  The plots
% include the TDE averages (avgY), the cumulative TDE, and the
% locations of the LPAs input by CALOC.  The initial value for
% the cumulative error is taken from the actual strip charts from
% CALOC.  The factor of 0.125 on the cumulative error is due to
% the frequency of the averages (i.e., 7.5 minutes is 1/8 of an
% hour).  The LPAs are plotted with a multiplicative factor of 3
% solely for ease of display.
figure
subplot(311), plot(xaxis, avgW, 'w+')
xlabel('Time (hours)'), ylabel('TDE (ns)')
hold on
plot(xaxis, -40 + cumsum(avgW * 0.125), 'w-')
plot(xaxis, plotlpaw * 3, 'wo')
grid on
axis([0 24 -80 80])
```

```
title('SEUS Whiskey 20Jan - Plot of TDE vs Time')

figure
subplot(311), plot(xaxis, avgX, 'w+')
xlabel('Time (hours)'), ylabel('TDE (ns)')
hold on
plot(xaxis, -60 + cumsum(avgX * 0.125), 'w-')
plot(xaxis, plotlpax * 3, 'wo')
grid on
axis([0 24 -80 80])
title('SEUS Xray 20Jan - Plot of TDE vs Time')

figure
subplot(311), plot(xaxis, avgY, 'w+')
xlabel('Time (hours)'), ylabel('TDE (ns)')
hold on
plot(xaxis, -50 + cumsum(avgY * 0.125), 'w-')
plot(xaxis, plotlpay * 3, 'wo')
grid on
axis([0 24 -80 80])
title('SEUS Yankee 20Jan - Plot of TDE vs Time')

figure
subplot(311), plot(xaxis, avgZ, 'w+')
xlabel('Time (hours)'), ylabel('TDE (ns)')
hold on
plot(xaxis, -92 + cumsum(avgZ * 0.125), 'w-')
plot(xaxis, plotlpaz * 3, 'wo')
grid on
axis([0 24 -80 80])
title('SEUS Zulu 20Jan - Plot of TDE vs Time')



================================================================================
================================================================================

% LORAN-C Thesis
%
% socus18.m
%
% Data Set:  Malone-SOCUS 18Jan97
```

```
% Program generates LORAN Time Difference Error (TDE) vs Time strip
% charts similar to the output of the current CALOC strip charts.
% These strip charts will be used to compare the output of the new
% control algorithm to determine the increase in control accuracy.

% Clear stored variables
clear

% Load data set
load socus18j.dat;

% Remove bias from data.  Original data has a range of 0-99 which
% equates to the actual range of -50 to +49.  Each single step
% increase in the data equates to an actual 10 ns increase in the
% TDE, therefore the data is scaled by 10.
socus18j = (socus18j - 50) * 10;

% Load LORAN chain data vectors for Victor, Whiskey, Xray, Yankee
% and Zulu.
socus18jV = socus18j(:,1);
socus18jW = socus18j(:,2);
socus18jX = socus18j(:,4);
socus18jY = socus18j(:,6);
socus18jZ = socus18j(:,9);

% Determine the size of the data file.
[num_rows, num_cols] = size(socus18j);

% Determine number of data points after taking 7.5 minute averages.
% Since each data point represents 10 seconds in real time, there
% are 45 data points in each 7.5 minute average.  Using the Matlab
% floor command rounds down the result of the operation to get an
% integer value to use in reshaping the data for averaging.  The
% second line generates the vector for plotting the data.
data_cols = floor(num_rows/45);
xaxis = [1:num_rows/45]/8;

% Take 7.5 minute averages to generate data points for plotting
% similar to current strip charts.  Averages are taken by reshaping
% the data into a matrix of dimensions data_cols x 45 and then taking
% the mean in a column-wise fashion.  This generates a vector of
% length data_cols of the 7.5 minute averages.
```

```
avgV = mean(reshape(socus18jV(1:(45*data_cols)), 45, data_cols));
avgW = mean(reshape(socus18jW(1:(45*data_cols)), 45, data_cols));
avgX = mean(reshape(socus18jX(1:(45*data_cols)), 45, data_cols));
avgY = mean(reshape(socus18jY(1:(45*data_cols)), 45, data_cols));
avgZ = mean(reshape(socus18jZ(1:(45*data_cols)), 45, data_cols));

% After plotting the raw data, the locations of the LPAs were
% determined by comparing the generated plot with the actual strip
% charts.  The locations are placed below to generate LPA vectors.
lpav = zeros(size(1:num_rows/45));
lpaw = zeros(size(1:num_rows/45));
lpax = zeros(size(1:num_rows/45));
lpay = zeros(size(1:num_rows/45));
lpaz = zeros(size(1:num_rows/45));

lpav(94) = -20;

lpav(60) = 20; lpav(132) = 20; lpav(154) = 20;

lpaw(39) = -20;

lpaw(64) = 20; lpaw(119) = 20; lpaw(134) = 20; lpaw(152) = 20;

lpax(24) = 20;

lpay(39) = -20; lpay(132) = -20;

lpay(26) = 20; lpay(30) = 20;

lpaz(111) = -20; lpaz(113) = -20; lpaz(130) = -40;

lpaz(12) = 20; lpaz(41) = 20; lpaz(65) = 20; lpaz(96) = 20;
lpaz(143) = 20; lpaz(163) = 20;

% Generate vector to plot LPA.  Dividing by zero generates a
% vector of NaN so that only the LPA locations determined in
% the for loop below will be plotted.
plotlpav = zeros(size(lpav))/0.0;
plotlpaw = zeros(size(lpaw))/0.0;
plotlpax = zeros(size(lpax))/0.0;
plotlpay = zeros(size(lpay))/0.0;
plotlpaz = zeros(size(lpaz))/0.0;
```

```
% This for loop determines the locations of the LPAs by indexing
% the capability of Matlab to index a vector based on its non-zero
% values.  The location of each LPA is transferred to the NaN
% vector generated above.  Since Matlab does not plot NaN, the only
% values that are plotted are the LPA locations.
for index=1:data_cols
   if lpav(index)~=0
      plotlpav(index)=lpav(index);
   end
   if lpaw(index)~=0
      plotlpaw(index)=lpaw(index);
   end
   if lpax(index)~=0
      plotlpax(index)=lpax(index);
   end
   if lpay(index)~=0
      plotlpay(index)=lpay(index);
   end
   if lpaz(index)~=0
      plotlpaz(index)=lpaz(index);
   end
end

% Generate the LORAN Strip Charts
%
% The plots are generated using subplot so that they appear with
% similar resolution to the existing strip charts.  The plots
% include the TDE averages (avgY), the cumulative TDE, and the
% locations of the LPAs input by CALOC.  The initial value for
% the cumulative error is taken from the actual strip charts from
% CALOC.  The factor of 0.125 on the cumulative error is due to
% the frequency of the averages (i.e., 7.5 minutes is 1/8 of an
% hour).  The LPAs are plotted with a multiplicative factor of 2
% or 3 solely for ease of display.
figure
subplot(311), plot(xaxis, avgV, 'w+')
xlabel('Time (hours)'), ylabel('TDE (ns)')
hold on
plot(xaxis, -55 + cumsum(avgV * 0.125), 'w-')
plot(xaxis, plotlpav * 3, 'wo')
grid on
```

```
axis([0 24 -80 80])
title('SOCUS Victor 18Jan - Plot of TDE vs Time')


figure
subplot(311), plot(xaxis, avgW, 'w+')
xlabel('Time (hours)'), ylabel('TDE (ns)')
hold on
plot(xaxis, -17 + cumsum(avgW * 0.125), 'w-')
plot(xaxis, plotlpaw * 3, 'wo')
grid on
axis([0 24 -80 80])
title('SOCUS Whiskey 18Jan - Plot of TDE vs Time')

figure
subplot(311), plot(xaxis, avgX, 'w+')
xlabel('Time (hours)'), ylabel('TDE (ns)')
hold on
plot(xaxis, 21 + cumsum(avgX * 0.125), 'w-')
plot(xaxis, plotlpax * 3, 'wo')
grid on
axis([0 24 -80 80])
title('SOCUS Xray 18Jan - Plot of TDE vs Time')

figure
subplot(311), plot(xaxis, avgY, 'w+')
xlabel('Time (hours)'), ylabel('TDE (ns)')
hold on
plot(xaxis, 16 + cumsum(avgY * 0.125), 'w-')
plot(xaxis, plotlpay * 3, 'wo')
grid on
axis([0 24 -80 80])
title('SOCUS Yankee 18Jan - Plot of TDE vs Time')

figure
subplot(311), plot(xaxis, avgZ, 'w+')
xlabel('Time (hours)'), ylabel('TDE (ns)')
hold on
plot(xaxis, -20 + cumsum(avgZ * 0.125), 'w-')
plot(xaxis, plotlpaz * 2, 'wo')
grid on
axis([0 24 -80 80])
```

```
title('SOCUS Zulu 18Jan - Plot of TDE vs Time')


================================================================================
================================================================================

% LORAN-C Thesis
%
% socus20.m
%
% Data Set:  Malone-SOCUS 20Jan97
% Program generates LORAN Time Difference Error (TDE) vs Time strip
% charts similar to the output of the current CALOC strip charts.
% These strip charts will be used to compare the output of the new
% control algorithm to determine the increase in control accuracy.

% Clear stored variables
clear

% Load data set
load socus20j.dat;

% Remove bias from data.  Original data has a range of 0-99 which
% equates to the actual range of -50 to +49.  Each single step
% increase in the data equates to an actual 10 ns increase in the
% TDE, therefore the data is scaled by 10.
socus20j = (socus20j - 50) * 10;

% Load LORAN chain data vectors for Victor, Whiskey, Xray, Yankee
% and Zulu.
socus20jV = socus20j(:,1);
socus20jW = socus20j(:,2);
socus20jX = socus20j(:,4);
socus20jY = socus20j(:,6);
socus20jZ = socus20j(:,9);

% Determine the size of the data file.
[num_rows, num_cols] = size(socus20j);

% Determine number of data points after taking 7.5 minute averages.
% Since each data point represents 10 seconds in real time, there
```

```
% are 45 data points in each 7.5 minute average.  Using the Matlab
% floor command rounds down the result of the operation to get an
% integer value to use in reshaping the data for averaging.  The
% second line generates the vector for plotting the data.
data_cols = floor(num_rows/45);
xaxis = [1:num_rows/45]/8;

% Take 7.5 minute averages to generate data points for plotting
% similar to current strip charts.  Averages are taken by reshaping
% the data into a matrix of dimensions data_cols x 45 and then taking
% the mean in a column-wise fashion.  This generates a vector of
% length data_cols of the 7.5 minute averages.
avgV = mean(reshape(socus20jV(1:(45*data_cols)), 45, data_cols));
avgW = mean(reshape(socus20jW(1:(45*data_cols)), 45, data_cols));
avgX = mean(reshape(socus20jX(1:(45*data_cols)), 45, data_cols));
avgY = mean(reshape(socus20jY(1:(45*data_cols)), 45, data_cols));
avgZ = mean(reshape(socus20jZ(1:(45*data_cols)), 54, data_cols));

% After plotting the raw data, the locations of the LPAs were
% determined by comparing the generated plot with the actual strip
% charts.  The locations are placed below to generate LPA vectors.
lpav = zeros(size(1:num_rows/45));
lpaw = zeros(size(1:num_rows/45));
lpax = zeros(size(1:num_rows/45));
lpay = zeros(size(1:num_rows/45));
lpaz = zeros(size(1:num_rows/45));

lpav(5) = 20; lpav(133) = 20;

lpaw(31) = -20; lpaw(69) = -20;

lpaw(9) = 20; lpaw(14) = 20; lpaw(87) = 20; lpaw(143) = 20;

lpax(61) = 20;

lpay(151) = -20;

lpay(64) = 20; lpay(79) = 20;

lpaz(105) = -20; lpaz(144) = -20; lpaz(158) = -20;

lpaz(9) = 20; lpaz(77) = 20; lpaz(91) = 20;
```

```
% Generate vector to plot LPA.  Dividing by zero generates a
% vector of NaN so that only the LPA locations determined in
% the for loop below will be plotted.
plotlpav = zeros(size(lpav))/0.0;
plotlpaw = zeros(size(lpaw))/0.0;
plotlpax = zeros(size(lpax))/0.0;
plotlpay = zeros(size(lpay))/0.0;
plotlpaz = zeros(size(lpaz))/0.0;

% This for loop determines the locations of the LPAs by indexing
% the capability of Matlab to index a vector based on its non-zero
% values.  The location of each LPA is transferred to the NaN
% vector generated above.  Since Matlab does not plot NaN, the only
% values that are plotted are the LPA locations.
for index=1:data_cols
   if lpav(index)~=0
      plotlpav(index)=lpav(index);
   end
   if lpaw(index)~=0
      plotlpaw(index)=lpaw(index);
   end
   if lpax(index)~=0
      plotlpax(index)=lpax(index);
   end
   if lpay(index)~=0
      plotlpay(index)=lpay(index);
   end
   if lpaz(index)~=0
      plotlpaz(index)=lpaz(index);
   end
end

% Generate the LORAN Strip Charts
%
% The plots are generated using subplot so that they appear with
% similar resolution to the existing strip charts.  The plots
% include the TDE averages (avgY), the cumulative TDE, and the
% locations of the LPAs input by CALOC.  The initial value for
% the cumulative error is taken from the actual strip charts from
% CALOC.  The factor of 0.125 on the cumulative error is due to
% the frequency of the averages (i.e., 7.5 minutes is 1/8 of an
```

```
% hour).  The LPAs are plotted with a multiplicative factor of 2
% or 3 solely for ease of display.
figure
subplot(311), plot(xaxis, avgV, 'w+')
xlabel('Time (hours)'), ylabel('TDE (ns)')
hold on
plot(xaxis, -42 + cumsum(avgV * 0.125), 'w-')
plot(xaxis, plotlpav * 3, 'wo')
grid on
axis([0 24 -80 80])
title('SOCUS Victor 20Jan - Plot of TDE vs Time')

figure
subplot(311), plot(xaxis, avgW, 'w+')
xlabel('Time (hours)'), ylabel('TDE (ns)')
hold on
plot(xaxis, -20 + cumsum(avgW * 0.125), 'w-')
plot(xaxis, plotlpaw * 3, 'wo')
grid on
axis([0 24 -80 80])
title('SOCUS Whiskey 20Jan - Plot of TDE vs Time')

figure
subplot(311), plot(xaxis, avgX, 'w+')
xlabel('Time (hours)'), ylabel('TDE (ns)')
hold on
plot(xaxis, 20 + cumsum(avgX * 0.125), 'w-')
plot(xaxis, plotlpax * 3, 'wo')
grid on
axis([0 24 -80 80])
title('SOCUS Xray 20Jan - Plot of TDE vs Time')

figure
subplot(311), plot(xaxis, avgY, 'w+')
xlabel('Time (hours)'), ylabel('TDE (ns)')
hold on
plot(xaxis, -42 + cumsum(avgY * 0.125), 'w-')
plot(xaxis, plotlpay * 3, 'wo')
grid on
axis([0 24 -80 80])
title('SOCUS Yankee 20Jan - Plot of TDE vs Time')
```

```
figure
subplot(311), plot(xaxis, avgZ, 'w+')
xlabel('Time (hours)'), ylabel('TDE (ns)')
hold on
plot(xaxis, 4 + cumsum(avgZ * 0.125), 'w-')
plot(xaxis, plotlpaz * 2, 'wo')
grid on
axis([0 24 -80 80])
title('SOCUS Zulu 20Jan - Plot of TDE vs Time')
```

```
================================================================================
================================================================================
```

```
% LORAN-C Thesis
%
% nocus4.m
%
% Data Set:  Middletown-NOCUS 4May97
% Program generates LORAN Time Difference Error (TDE) vs Time strip
% charts similar to the output of the current CALOC strip charts.
% These strip charts will be used to compare the output of the new
% control algorithm to determine the increase in control accuracy.

% Clear stored variables
clear

% Load data set
load nocus4my.dat;

% Remove bias from data.  Original data has a range of 0-99 which
% equates to the actual range of -50 to +49.  Each single step
% increase in the data equates to an actual 10 ns increase in the
% TDE, therefore the data is scaled by 10.
nocus4my = (nocus4my - 50) * 10;

% Load LORAN chain data vectors for Whiskey, Xray and Yankee.
nocus4myW = nocus4my(:,10);
nocus4myX = nocus4my(:,12);
nocus4myY = nocus4my(:,3);
```

```
% Determine the size of the data file.
[num_rows, num_cols] = size(nocus4my);

% Determine number of data points after taking 7.5 minute averages.
% Since each data point represents 10 seconds in real time, there
% are 45 data points in each 7.5 minute average.  Using the Matlab
% floor command rounds down the result of the operation to get an
% integer value to use in reshaping the data for averaging.  The
% second line generates the vector for plotting the data.
data_cols = floor(num_rows/45);
xaxis = [1:num_rows/45]/8;

% Take 7.5 minute averages to generate data points for plotting
% similar to current strip charts.  Averages are taken by reshaping
% the data into a matrix of dimensions data_cols x 45 and then taking
% the mean in a column-wise fashion.  This generates a vector of
% length data_cols of the 7.5 minute averages.
avgW = mean(reshape(nocus4myW(1:(45*data_cols)), 45, data_cols));
avgX = mean(reshape(nocus4myX(1:(45*data_cols)), 45, data_cols));
avgY = mean(reshape(nocus4myY(1:(45*data_cols)), 45, data_cols));

% After plotting the raw data, the locations of the LPAs were
% determined by comparing the generated plot with the actual strip
% charts.  The locations are placed below to generate LPA vectors.
lpaw = zeros(size(1:num_rows/45));
lpax = zeros(size(1:num_rows/45));
lpay = zeros(size(1:num_rows/45));

lpaw(108) = -20; lpaw(133) = -20;
lpaw(157) = -20; lpaw(178) = -20;

lpaw(13) = 40; lpaw(24) = 20; lpaw(79) = 20;

lpax(145) = -20; lpax(157) = -20;

lpax(48) = 20; lpax(178) = 20;

lpay(126) = -20; lpay(146) = -20;
lpay(158) = -20; lpay(178) = -20;

lpay(39) = 20; lpay(55) = 20; lpay(60) = 40;
```

```
% Generate vector to plot LPA.  Dividing by zero generates a
% vector of NaN so that only the LPA locations determined in
% the for loop below will be plotted.
plotlpaw = zeros(size(lpaw))/0.0;
plotlpax = zeros(size(lpax))/0.0;
plotlpay = zeros(size(lpay))/0.0;

% This for loop determines the locations of the LPAs by indexing
% the capability of Matlab to index a vector based on its non-zero
% values.  The location of each LPA is transferred to the NaN
% vector generated above.  Since Matlab does not plot NaN, the only
% values that are plotted are the LPA locations.
for index = 1:data_cols
   if lpaw(index)~=0
      plotlpaw(index)=lpaw(index);
   end
   if lpax(index)~=0
      plotlpax(index)=lpax(index);
   end
   if lpay(index)~=0
      plotlpay(index)=lpay(index);
   end
end

% Generate the LORAN Strip Charts
%
% The plots are generated using subplot so that they appear with
% similar resolution to the existing strip charts.  The plots
% include the TDE averages (avgY), the cumulative TDE, and the
% locations of the LPAs input by CALOC.  The initial value for
% the cumulative error is taken from the actual strip charts from
% CALOC.  The factor of 0.125 on the cumulative error is due to
% the frequency of the averages (i.e., 7.5 minutes is 1/8 of an
% hour).  The LPAs are plotted with a multiplicative factor of 3
% solely for ease of display.
figure
subplot(311), plot(xaxis, avgW, 'w+')
xlabel('Time (hours)'), ylabel('TDE (ns)')
hold on
plot(xaxis, -35 + cumsum(avgW * 0.125), 'w-')
plot(xaxis, plotlpaw * 3, 'wo')
grid on
```

```
axis([0 24 -80 80])
title('NOCUS Whiskey 4May - Plot of TDE vs Time')

figure
subplot(311), plot(xaxis, avgX, 'w+')
xlabel('Time (hours)'), ylabel('TDE (ns)')
hold on
plot(xaxis, 30 + cumsum(avgX * 0.125), 'w-')
plot(xaxis, plotlpax * 3, 'wo')
grid on
axis([0 24 -80 80])
title('NOCUS Xray 4May - Plot of TDE vs Time')

figure
subplot(311), plot(xaxis, avgY, 'w+')
xlabel('Time (hours)'), ylabel('TDE (ns)')
hold on
plot(xaxis, 25 + cumsum(avgY * 0.125), 'w-')
plot(xaxis, plotlpay * 3, 'wo')
grid on
axis([0 24 -80 80])
title('NOCUS Yankee 4May - Plot of TDE vs Time')
```

```
==========================================================================
==========================================================================

% LORAN-C Thesis
%
% nocus5.m
%
% Data Set:  Middletown-NOCUS 5May97
% Program generates LORAN Time Difference Error (TDE) vs Time strip
% charts similar to the output of the current CALOC strip charts.
% These strip charts will be used to compare the output of the new
% control algorithm to determine the increase in control accuracy.

% Clear stored variables
clear

% Load data set
```

```
load nocus5my.dat;

% Remove bias from data.  Original data has a range of 0-99 which
% equates to the actual range of -50 to +49.  Each single step
% increase in the data equates to an actual 10 ns increase in the
% TDE, therefore the data is scaled by 10.
nocus5my = (nocus5my - 50) * 10;

% Load LORAN chain data vectors for Whiskey, Xray and Yankee.
nocus5myW = nocus5my(:,10);
nocus5myX = nocus5my(:,12);
nocus5myY = nocus5my(:,3);

% Determine the size of the data file.
[num_rows, num_cols] = size(nocus5my);

% Determine number of data points after taking 7.5 minute averages.
% Since each data point represents 10 seconds in real time, there
% are 45 data points in each 7.5 minute average.  Using the Matlab
% floor command rounds down the result of the operation to get an
% integer value to use in reshaping the data for averaging.  The
% second line generates the vector for plotting the data.
data_cols = floor( num_rows/45 );
xaxis = [1:num_rows/45]/8;

% Take 7.5 minute averages to generate data points for plotting
% similar to current strip charts.  Averages are taken by reshaping
% the data into a matrix of dimensions data_cols x 45 and then taking
% the mean in a column-wise fashion.  This generates a vector of
% length data_cols of the 7.5 minute averages.
avgW = mean(reshape(nocus5myW(1:(45*data_cols)), 45, data_cols));
avgX = mean(reshape(nocus5myX(1:(45*data_cols)), 45, data_cols));
avgY = mean(reshape(nocus5myY(1:(45*data_cols)), 45, data_cols));

% After plotting the raw data, the locations of the LPAs were
% determined by comparing the generated plot with the actual strip
% charts.  The locations are placed below to generate LPA vectors.
lpaw = zeros(size(1:num_rows/45));
lpax = zeros(size(1:num_rows/45));
lpay = zeros(size(1:num_rows/45));

lpaw(147) = -20;
```

```
lpax(105) = -20;

lpax(12) = 20;

lpay(115) = -20; lpay(147) = -40;

lpay(13) = 20; lpay(30) = 20;
lpay(50) = 20; lpay(74) = 20;

% Generate vector to plot LPA.  Dividing by zero generates a
% vector of NaN so that only the LPA locations determined in
% the for loop below will be plotted.
plotlpaw = zeros(size(lpaw))/0.0;
plotlpax = zeros(size(lpax))/0.0;
plotlpay = zeros(size(lpay))/0.0;

% This for loop determines the locations of the LPAs by indexing
% the capability of Matlab to index a vector based on its non-zero
% values.  The location of each LPA is transferred to the NaN
% vector generated above.  Since Matlab does not plot NaN, the only
% values that are plotted are the LPA locations.
for index = 1:data_cols
   if lpaw(index)~=0
      plotlpaw(index)=lpaw(index);
   end
   if lpax(index)~=0
      plotlpax(index)=lpax(index);
   end
   if lpay(index)~=0
      plotlpay(index)=lpay(index);
   end
end

% Generate the LORAN Strip Charts
%
% The plots are generated using subplot so that they appear with
% similar resolution to the existing strip charts.  The plots
% include the TDE averages (avgY), the cumulative TDE, and the
% locations of the LPAs input by CALOC.  The initial value for
% the cumulative error is taken from the actual strip charts from
% CALOC.  The factor of 0.125 on the cumulative error is due to
```

```
% the frequency of the averages (i.e., 7.5 minutes is 1/8 of an
% hour).  The LPAs are plotted with a multiplicative factor of 3
% solely for ease of display.
figure
subplot(311), plot(xaxis, avgW, 'w+')
xlabel('Time (hours)'), ylabel('TDE (ns)')
hold on
plot(xaxis, -15 + cumsum(avgW * 0.125), 'w-')
plot(xaxis, plotlpaw * 3, 'wo')
grid on
axis([0 24 -80 80])
title('NOCUS Whiskey 5May - Plot of TDE vs Time')

figure
subplot(311), plot(xaxis, avgX, 'w+')
xlabel('Time (hours)'), ylabel('TDE (ns)')
hold on
plot(xaxis, -25 + cumsum(avgX * 0.125), 'w-')
plot(xaxis, plotlpax * 3, 'wo')
grid on
axis([0 24 -80 80])
title('NOCUS Xray 5May - Plot of TDE vs Time')

figure
subplot(311), plot(xaxis, avgY, 'w+')
xlabel('Time (hours)'), ylabel('TDE (ns)')
hold on
plot(xaxis, 5 + cumsum(avgY * 0.125), 'w-')
plot(xaxis, plotlpay * 3, 'wo')
grid on
axis([0 24 -80 80])
title('NOCUS Yankee 5May - Plot of TDE vs Time')



================================================================================
================================================================================

% LORAN-C Thesis
%
% uswc4.m
%
```

```
% Data Set:  USWC 4May97
% Program generates LORAN Time Difference Error (TDE) vs Time strip
% charts similar to the output of the current CALOC strip charts.
% These strip charts will be used to compare the output of the new
% control algorithm to determine the increase in control accuracy.

% Clear stored variables
clear

% Load data set
load uswc4my.dat;

% Remove bias from data.  Original data has a range of 0-99 which
% equates to the actual range of -50 to +49.  Each single step
% increase in the data equates to an actual 10 ns increase in the
% TDE, therefore the data is scaled by 10.
uswc4my = (uswc4my - 50) * 10;

% Load LORAN chain data vectors for Whiskey, Xray and Yankee.
uswc4myW = uswc4my(:,1);
uswc4myX = uswc4my(:,10);
uswc4myY = uswc4my(:,12);

% Determine the size of the data file.
[num_rows, num_cols] = size(uswc4my);

% Determine number of data points after taking 7.5 minute averages.
% Since each data point represents 10 seconds in real time, there
% are 45 data points in each 7.5 minute average.  Using the Matlab
% floor command rounds down the result of the operation to get an
% integer value to use in reshaping the data for averaging.  The
% second line generates the vector for plotting the data.
data_cols = floor( num_rows/45 );
xaxis = [1:num_rows/45]/8;

% Take 7.5 minute averages to generate data points for plotting
% similar to current strip charts.  Averages are taken by reshaping
% the data into a matrix of dimensions data_cols x 45 and then taking
% the mean in a column-wise fashion.  This generates a vector of
% length data_cols of the 7.5 minute averages.
avgW = mean(reshape(uswc4myW(1:(45*data_cols)), 45, data_cols));
avgX = mean(reshape(uswc4myX(1:(45*data_cols)), 45, data_cols));
```

```
avgY = mean(reshape(uswc4myY(1:(45*data_cols)), 45, data_cols));

% After plotting the raw data, the locations of the LPAs were
% determined by comparing the generated plot with the actual strip
% charts.  The locations are placed below to generate LPA vectors.
lpaw = zeros(size(1:num_rows/45));
lpax = zeros(size(1:num_rows/45));
lpay = zeros(size(1:num_rows/45));

lpaw(5) = -20; lpaw(45) = -20;
lpaw(50) = -20; lpaw(84) = -20;

lpaw(94) = 20; lpaw(104) = 20;

lpax(69) = -20;

lpax(34) = 20; lpax(50) = 20; lpax(154) = 20;

lpay(84) = -20;

lpay(40) = 20; lpay(55) = 20;

% Generate vector to plot LPA.  Dividing by zero generates a
% vector of NaN so that only the LPA locations determined in
% the for loop below will be plotted.
plotlpaw = zeros(size(lpaw))/0.0;
plotlpax = zeros(size(lpax))/0.0;
plotlpay = zeros(size(lpay))/0.0;

% This for loop determines the locations of the LPAs by indexing
% the capability of Matlab to index a vector based on its non-zero
% values.  The location of each LPA is transferred to the NaN
% vector generated above.  Since Matlab does not plot NaN, the only
% values that are plotted are the LPA locations.
for index = 1:data_cols
   if lpaw(index)~=0
      plotlpaw(index)=lpaw(index);
   end
   if lpax(index)~=0
      plotlpax(index)=lpax(index);
   end
   if lpay(index)~=0
```

```
      plotlpay(index)=lpay(index);
   end
end

% Generate the LORAN Strip Charts
%
% The plots are generated using subplot so that they appear with
% similar resolution to the existing strip charts.  The plots
% include the TDE averages (avgY), the cumulative TDE, and the
% locations of the LPAs input by CALOC.  The initial value for
% the cumulative error is taken from the actual strip charts from
% CALOC.  The factor of 0.125 on the cumulative error is due to
% the frequency of the averages (i.e., 7.5 minutes is 1/8 of an
% hour).  The LPAs are plotted with a multiplicative factor of 3
% solely for ease of display.
figure
subplot(311), plot(xaxis, avgW, 'w+')
xlabel('Time (hours)'), ylabel('TDE (ns)')
hold on
plot(xaxis, 80 + cumsum(avgW * 0.125), 'w-')
plot(xaxis, plotlpaw * 3, 'wo')
grid on
axis([0 24 -80 80])
title('USWC Whiskey 4May - Plot of TDE vs Time')

figure
subplot(311), plot(xaxis, avgX, 'w+')
xlabel('Time (hours)'), ylabel('TDE (ns)')
hold on
plot(xaxis, -30 + cumsum(avgX * 0.125), 'w-')
plot(xaxis, plotlpax * 3, 'wo')
grid on
axis([0 24 -80 80])
title('USWC Xray 4May - Plot of TDE vs Time')

figure
subplot(311), plot(xaxis, avgY, 'w+')
xlabel('Time (hours)'), ylabel('TDE (ns)')
hold on
plot(xaxis, -40 + cumsum(avgY * 0.125), 'w-')
plot(xaxis, plotlpay * 3, 'wo')
grid on
```

```
axis([0 24 -80 80])
title('USWC Yankee 4May - Plot of TDE vs Time')



================================================================================
================================================================================

% LORAN-C Thesis
%
% uswc5.m
%
% Data Set:  Middletown-USWC 5May97
% Program generates LORAN Time Difference Error (TDE) vs Time strip
% charts similar to the output of the current CALOC strip charts.
% These strip charts will be used to compare the output of the new
% control algorithm to determine the increase in control accuracy.

% Clear stored variables
clear

% Load data set
load uswc5my.dat;

% Remove bias from data.  Original data has a range of 0-99 which
% equates to the actual range of -50 to +49.  Each single step
% increase in the data equates to an actual 10 ns increase in the
% TDE, therefore the data is scaled by 10.
uswc5my = (uswc5my - 50) * 10;

% Load LORAN chain data vectors for Whiskey, Xray and Yankee.
uswc5myW = uswc5my(:,1);
uswc5myX = uswc5my(:,10);
uswc5myY = uswc5my(:,12);

% Determine the size of the data file.
[num_rows, num_cols] = size(uswc5my);

% Determine number of data points after taking 7.5 minute averages.
% Since each data point represents 10 seconds in real time, there
% are 45 data points in each 7.5 minute average.  Using the Matlab
% floor command rounds down the result of the operation to get an
```

```
% integer value to use in reshaping the data for averaging.  The
% second line generates the vector for plotting the data.
data_cols = floor( num_rows/45 );
xaxis = [1:num_rows/45]/8;

% Take 7.5 minute averages to generate data points for plotting
% similar to current strip charts.  Averages are taken by reshaping
% the data into a matrix of dimensions data_cols x 45 and then taking
% the mean in a column-wise fashion.  This generates a vector of
% length data_cols of the 7.5 minute averages.
avgW = mean(reshape(uswc5myW(1:(45*data_cols)), 45, data_cols));
avgX = mean(reshape(uswc5myX(1:(45*data_cols)), 45, data_cols));
avgY = mean(reshape(uswc5myY(1:(45*data_cols)), 45, data_cols));

% After plotting the raw data, the locations of the LPAs were
% determined by comparing the generated plot with the actual strip
% charts.  The locations are placed below to generate LPA vectors.
lpaw = zeros(size(1:num_rows/45));
lpax = zeros(size(1:num_rows/45));
lpay = zeros(size(1:num_rows/45));

lpaw(45) = -20; lpaw(61) = -20;
lpaw(71) = -20;

lpaw(94) = 20; lpaw(121) = 20;

lpax(75) = -40;

lpax(33) = 20; lpax(81) = 20; lpax(94) = 20;

lpay(75) = -40;

lpay(48) = 20; lpay(94) = 20; lpay(157) = 20;

% Generate vector to plot LPA.  Dividing by zero generates a
% vector of NaN so that only the LPA locations determined in
% the for loop below will be plotted.
plotlpaw = zeros(size(lpaw))/0.0;
plotlpax = zeros(size(lpax))/0.0;
plotlpay = zeros(size(lpay))/0.0;

% This for loop determines the locations of the LPAs by indexing
```

```
% the capability of Matlab to index a vector based on its non-zero
% values.  The location of each LPA is transferred to the NaN
% vector generated above.  Since Matlab does not plot NaN, the only
% values that are plotted are the LPA locations.
for index = 1:data_cols
   if lpaw(index)~=0
      plotlpaw(index)=lpaw(index);
   end
   if lpax(index)~=0
      plotlpax(index)=lpax(index);
   end
   if lpay(index)~=0
      plotlpay(index)=lpay(index);
   end
end

% Generate the LORAN Strip Charts
%
% The plots are generated using subplot so that they appear with
% similar resolution to the existing strip charts.  The plots
% include the TDE averages (avgY), the cumulative TDE, and the
% locations of the LPAs input by CALOC.  The initial value for
% the cumulative error is taken from the actual strip charts from
% CALOC.  The factor of 0.125 on the cumulative error is due to
% the frequency of the averages (i.e., 7.5 minutes is 1/8 of an
% hour).  The LPAs are plotted with a multiplicative factor of 2
% or 3 solely for ease of display.
figure
subplot(311), plot(xaxis, avgW, 'w+')
xlabel('Time (hours)'), ylabel('TDE (ns)')
hold on
plot(xaxis, 60 + cumsum(avgW * 0.125), 'w-')
plot(xaxis, plotlpaw * 3, 'wo')
grid on
axis([0 24 -80 80])
title('USWC Whiskey 5May - Plot of TDE vs Time')

figure
subplot(311), plot(xaxis, avgX, 'w+')
xlabel('Time (hours)'), ylabel('TDE (ns)')
hold on
plot(xaxis, 25 + cumsum(avgX * 0.125), 'w-')
```

```
plot(xaxis, plotlpax * 2, 'wo')
grid on
axis([0 24 -80 80])
title('USWC Xray 5May - Plot of TDE vs Time')

figure
subplot(311), plot(xaxis, avgY, 'w+')
xlabel('Time (hours)'), ylabel('TDE (ns)')
hold on
plot(xaxis, 25 + cumsum(avgY * 0.125), 'w-')
plot(xaxis, plotlpay * 2, 'wo')
grid on
axis([0 24 -80 80])
title('USWC Yankee 5May - Plot of TDE vs Time')



================================================================
================================================================

% LORAN-C Thesis
%
% kodiakzulu.m
%
% Data Set:  Kodiak Zulu
% Program generates LORAN Time Difference Error (TDE) vs Time strip
% charts similar to the output of the current CALOC strip charts.
% These strip charts will be used to compare the output of the new
% control algorithm to determine the increase in control accuracy.

% Clear stored variables
clear

% Load data set
load kodiak.dat;

% Remove bias from data.  Original data has a range of 0-99 which
% equates to the actual range of -50 to +49.  Each single step
% increase in the data equates to an actual 10 ns increase in the
% TDE, therefore the data is scaled by 10.
kodiak = (kodiak - 50) * 10;
```

```
% Load individual LORAN chain data vector for Zulu.
kodiakZ = kodiak(:,1);

% Determine the size of the data file.
[num_rows, num_cols] = size(kodiak);

% Determine number of data points after taking 7.5 minute averages.
% Since each data point represents 10 seconds in real time, there
% are 45 data points in each 7.5 minute average.  Using the Matlab
% floor command rounds down the result of the operation to get an
% integer value to use in reshaping the data for averaging.  The
% second line generates the vector for plotting the data.
data_cols = floor( num_rows/45 );
xaxis = [1:num_rows/45]/8;

% Take 7.5 minute averages to generate data points for plotting
% similar to current strip charts.  Averages are taken by reshaping
% the data into a matrix of dimensions data_cols x 45 and then taking
% the mean in a column-wise fashion.  This generates a vector of
% length data_cols of the 7.5 minute averages.
avgZ = mean(reshape(kodiakZ(1:(45*data_cols)), 45, data_cols));

% After plotting the raw data, the locations of the LPAs were
% determined by comparing the generated plot with the actual strip
% charts.  The locations are placed below to generate LPA vectors.
lpaz = zeros(size(1:data_cols));

lpaz(37) = -20; lpaz(82) = -20;
lpaz(9) = 20; lpaz(46) = 20;

% Generate vector to plot LPA.  Dividing by zero generates a
% vector of NaN so that only the LPA locations determined in
% the for loop below will be plotted.
plotlpaz = zeros(size(lpaz))/0.0;

% This for loop determines the locations of the LPAs by indexing
% the capability of Matlab to index a vector based on its non-zero
% values.  The location of each LPA is transferred to the NaN
% vector generated above.  Since Matlab does not plot NaN, the only
% values that are plotted are the LPA locations.
for index=1:data_cols
   if lpaz(index)~=0
```

```
      plotlpaz(index)=lpaz(index);
   end
end

% Generate the LORAN Strip Charts
%
% The plots are generated using subplot so that they appear with
% similar resolution to the existing strip charts.  The plots
% include the TDE averages (avgY), the cumulative TDE, and the
% locations of the LPAs input by CALOC.  The initial value for
% the cumulative error is taken from the actual strip charts from
% CALOC.  The factor of 0.125 on the cumulative error is due to
% the frequency of the averages (i.e., 7.5 minutes is 1/8 of an
% hour).  The LPAs are plotted with a multiplicative factor of 3
% solely for ease of display.
subplot(311), plot(xaxis, avgZ, 'w+')
xlabel('Time (hours)'), ylabel('TDE (ns)')
hold on
plot(xaxis, -15 + cumsum(avgZ * 0.125), 'w-')
plot(xaxis, plotlpaz * 3, 'wo')
grid on, axis([0 24 -80 80])
title('Kodiak Zulu - Plot of TDE vs Time')
```

7. PARSE TITLE FUNCTION

```
=========================================================================
=========================================================================

function graph_title = parse_title(string)
% Loran-C Thesis
%
% PARSE_TITLE  This function parses the data file name string and
%              generates a string for the title of the output graph.

% Get Loran-C chain name
if strcmp(string(1:5), 'seus_')
   chain = 'SEUS';
elseif strcmp(string(1:5), 'neus_')
   chain = 'NEUS';
elseif strcmp(string(1:5), 'socus')
   chain = 'SOCUS';
elseif strcmp(string(1:5), 'nocus')
   chain = 'NOCUS';
elseif strcmp(string(1:5), 'uswc_')
   chain = 'USWC';
elseif strcmp(string(1:5), 'kodik')
   chain = 'KODIAK';
end

% Get date
date = string(6:7);

% Get month
if strcmp(string(8:9), 'jn')
   month = 'January';
elseif strcmp(string(8:9), 'fb')
   month = 'February';
elseif strcmp(string(8:9), 'mr')
   month = 'March';
elseif strcmp(string(8:9), 'ar')
   month = 'April';
elseif strcmp(string(8:9), 'my')
   month = 'May';
elseif strcmp(string(8:9), 'ju')
   month = 'June';
```

```
elseif strcmp(string(8:9), 'jl')
   month = 'July';
elseif strcmp(string(8:9), 'ag')
   month = 'August';
elseif strcmp(string(8:9), 'sp')
   month = 'September';
elseif strcmp(string(8:9), 'ot')
   month = 'October';
elseif strcmp(string(8:9), 'nv')
   month = 'November';
elseif strcmp(string(8:9), 'dc')
   month = 'December';
end

% Get Loran-C station designator
if strcmp(string(10), 'V')
   station = 'Victor';
elseif strcmp(string(10), 'W')
   station = 'Whiskey';
elseif strcmp(string(10), 'X')
   station = 'Xray';
elseif strcmp(string(10), 'Y')
   station = 'Yankee';
elseif strcmp(string(10), 'Z')
   station = 'Zulu';
end

graph_title = [chain, ' ', station, ' (', date, ' ', month, ')'];
```

LIST OF REFERENCES

[1] "Loran-C User Handbook COMDTPUB P16562.6," U.S. Department of Transportation, U.S. Coast Guard, Commandant (G-NRN-1), 2100 Second Street SW, Washington, DC, 1992.

[2] "Project W1234, Consolidated Control," United States Coast Guard, Electronics Engineering Center, Wildwood, NJ, June, 1995.

[3] LCDR D. S. Taggert, "Project 4E3-0017.9W, Loran-C EERP Systems Analysis and Standards, Interim Report No. 3," United States Coast Guard, Electronics Engineering Center, Wildwood, NJ, May, 1990.

[4] D. Feldman, "On the Modeling, Estimation and Control of LORAN-C Time Differences," United States Coast Guard, Electronics Engineering Center, Wildwood, NJ, February, 1975.

[5] "Loran-C Engineering Course," Department of Transportation, United States Coast Guard, New London, CT, 1995.

[6] Dean C. Bruckner, "Automatic Pulse Shaping with the AN/FPN-42 and AN/FPN-44A Loran-C Transmitters," Master's Thesis, Naval Postgraduate School, Monterey, CA, 1993.

[7] Doug Taggert and Ben Stewart, "Electronic Equipment Replacement Project (EERP)," United States Coast Guard, Electronics Engineering Center, Wildwood, NJ, 1989.

[8] J. T. Doherty and D. A. Feldman, "Calculator Assisted LORAN-C Controller for Time Difference Error Control," Proceedings 5th Annual Technical Symposium, pages 49-55, Wild Goose Association, 1976.

[9] "Specification of the Transmitted Loran-C Signal COMDTINST M16562.4A," U.S. Department of Transportation, U.S. Coast Guard, Commandant (G-NRN-1), 2100 Second Street SW, Washington, DC, 1994.

[10] John G. Proakis and Dimitris G. Manolakis, *Digital Signal Processing: Principles, Algorithms, and Applications*, MacMillan Publishing Company, NY, 1993.

[11] Chi-Tsong Chen, *Analog and Digital Control System Design: Transfer-Function, State-Space, and Algebraic Methods*, Saunders College Publishing, NY, 1992.

[12] Robert H. Doherty and J. Ralph Johler, "Meteorological Influences on Loran-C Ground Wave Propagation," United States Department of Commerce, Office of Telecommunications, Institute for Telecommuncations, Boulder, CO, September, 1973.

[13] M. L. Meeks, *Radar Propagation at Low Altitudes*, Artech House, Inc., Dedham, MA, 1982.

[14] B. B. Peterson and R. J. Hartnett, "Measurement Techniques for Narrowband Interference to LORAN," Center for Advanced Studies, United States Coast Guard Academy, New London, CT, October, 1990.

[15] Simon Haykin, *Adaptive Filter Theory*, Prentice Hall, Inc., Englewood Cliffs, NJ, 2nd Edition, 1991.

INITIAL DISTRIBUTION LIST

1. Defense Technical Information Center 2
   8725 John J. Kingman Rd., Ste 0944
   Ft. Belvoir, VA 22060-6218

2. Dudley Knox Library 2
   Naval Postgraduate School
   411 Dyer Rd.
   Monterey, CA 93943-5101

3. Chairman, Code EC 1
   Department of Electrical and Computer Engineering
   Naval Postgraduate School
   Monterey, CA 93943-5121

4. Prof. Murali Tummala, Code EC/Tu 4
   Department of Electrical and Computer Engineering
   Naval Postgraduate School
   Monterey, CA 93943-5121

5. Prof. Roberto Cristi, Code EC/Cx 1
   Department of Electrical and Computer Engineering
   Naval Postgraduate School
   Monterey, CA 93943-5121

6. LCDR Chuck Schue 1
   U.S. Coast Guard
   Loran Support Unit
   12001 Pacific Avenue
   Wildwood, NJ 08260-3232

7. LT Steve Bartlett 1
   U.S. Coast Guard
   Loran Support Unit
   12001 Pacific Avenue
   Wildwood, NJ 08260-3232

8. LT Frederick M. France, Jr., USN 3
   1048 Spruance Road
   Monterey, CA 93940-4821